Prognostic Enhancements to Diagnostic Systems

Impact Technologies, LLC
Phase II SBIR

Contract Number: N00167-01-C-0064

Michael J. Roemer, Ph.D.

Impact Technologies, LLC
125 Tech Park Drive
Rochester, NY 14623

716-424-1990

mike.roemer@impact-tek.com

Carl Byington, P.E.
Matthew Watson
Kurt Grieb

Impact Technologies, LLC
220 Regent Court
State College, PA 16801
814-861-6273

carl.byington@impact-tek.com

Mitchell Lebold

Penn State Applied Research Laboratory
Condition-based Maintenance Department
814-865-8958

lebold@psu.edu

Agenda

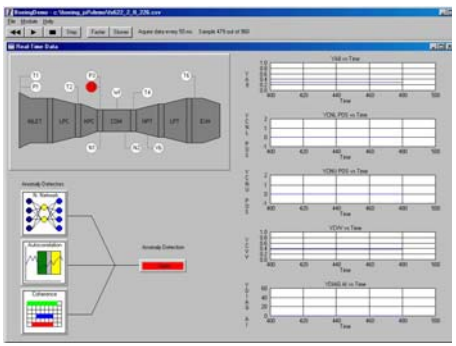
0830-0835	Introductions	Galie
0835-0900	PEDS Vision and Program Objectives	Byington, Roemer
0900-0930	Phase II Objectives, Workplan, Schedule	Byington
0930-0950	OSA and Prognostics Architecture Development	Byington, Watson
0950-1020	Specific Module Development Status • Power Transmission, Gas Turbine, Diesel	Byington, Watson
1020-1030	Break	All
1030-1050	Module Integration and Demonstration • ICAS, GT CBM, FADEC	Byington, Watson
1050-1110	Deliverable Status and Near-Term Milestones • MARCON Session and Paper • ICAS Integration & Transition Planning	Byington
1110-1200	Open Discussion and Action Items	All
1200	Adjourn	

PEDS Vision and Program Objectives

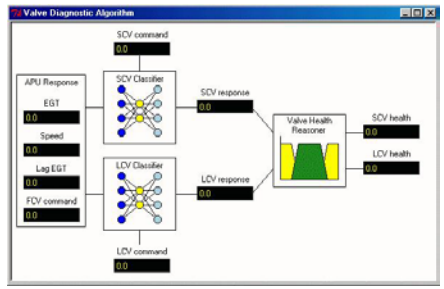
Impact Technologies Core Expertise

Maintenance Planning

Real-Time Health Monitoring



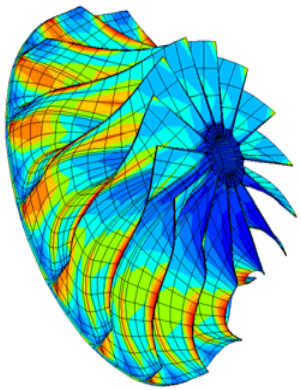
Diagnostics / Prognostics



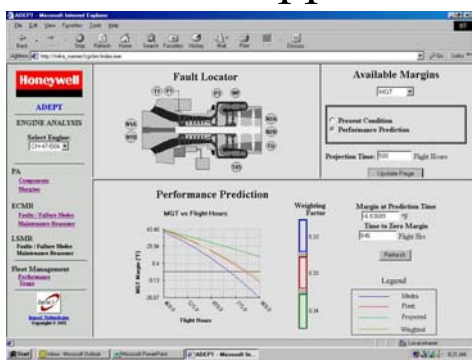
IMPACT Technologies LLC

Life-Cycle Cost Reduction

Engineering Analysis and Design



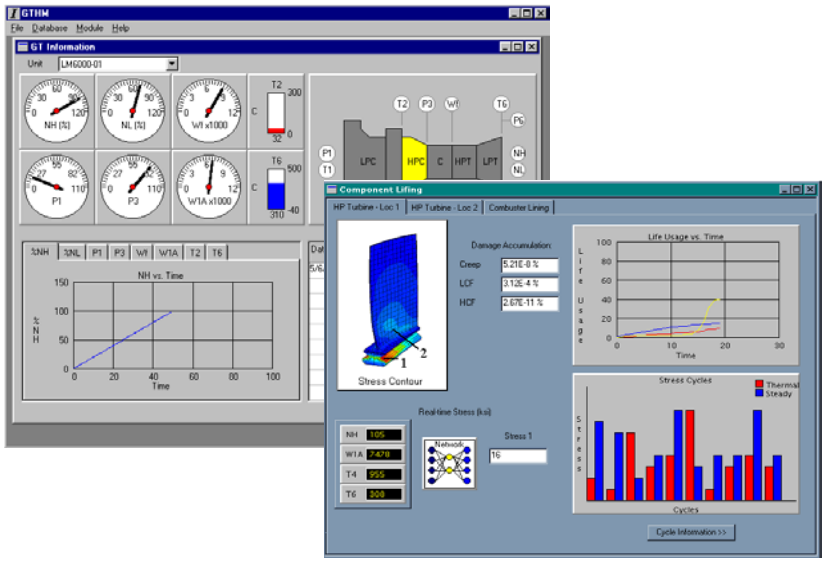
Web-Based Applications

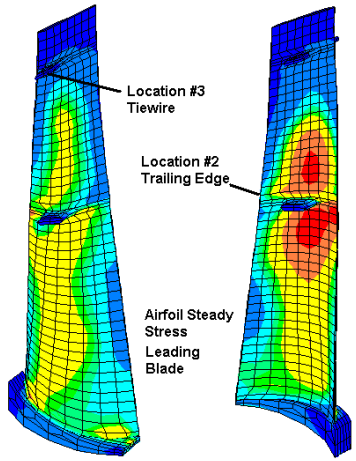


Performance, Vibration & Structural Testing

Impact's Commercial Products

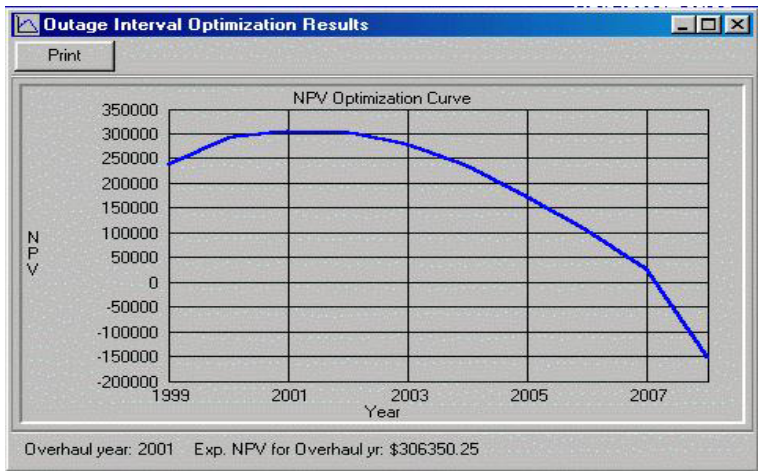
GT-Health Monitor



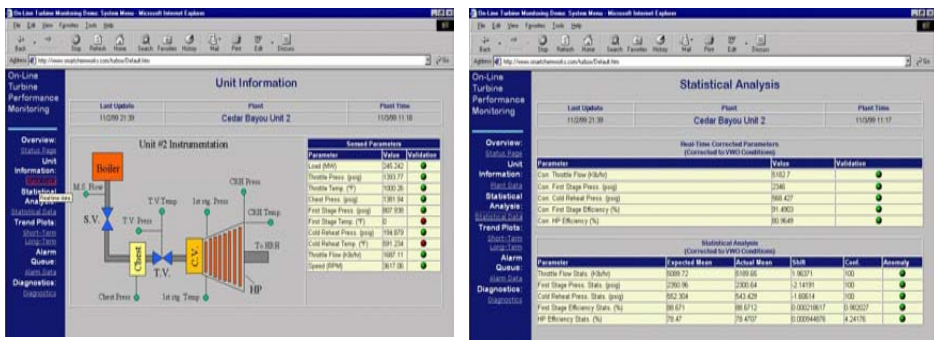


BladePro
And
ImPro
FE
Software

Turbo-X Maintenance Optimization



Turbo-Health Web Monitor



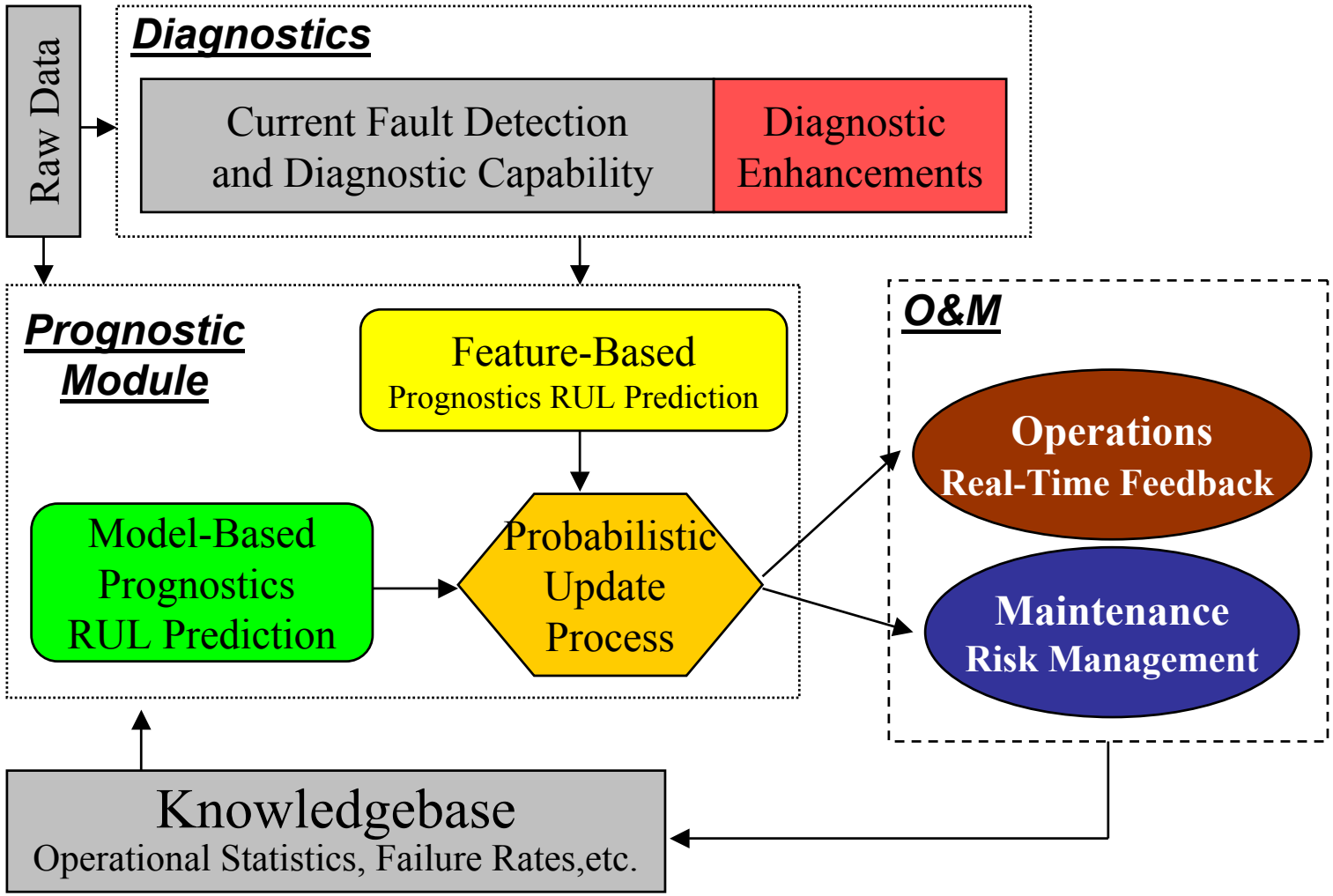
PEDS Vision Statements

- Develop machinery data analysis and software engineering tools for designing material condition prognostic modules.
- Deliver working prognostic software modules for specific types of maintenance events on specific components of Gas Turbine or Diesel engines and power transmission systems.
- Install software modules either on a marine Gas Turbine system's shipboard central condition monitoring system (i.e., ICAS) or a local microprocessor based controller or distributed between the local controller and central monitor.
- Demonstrate a process for designing families of machinery condition prognostic software modules.
- Deliver initial prognostic software modules, software design specifications and advanced data analysis algorithms at the conclusion of the Phase II.

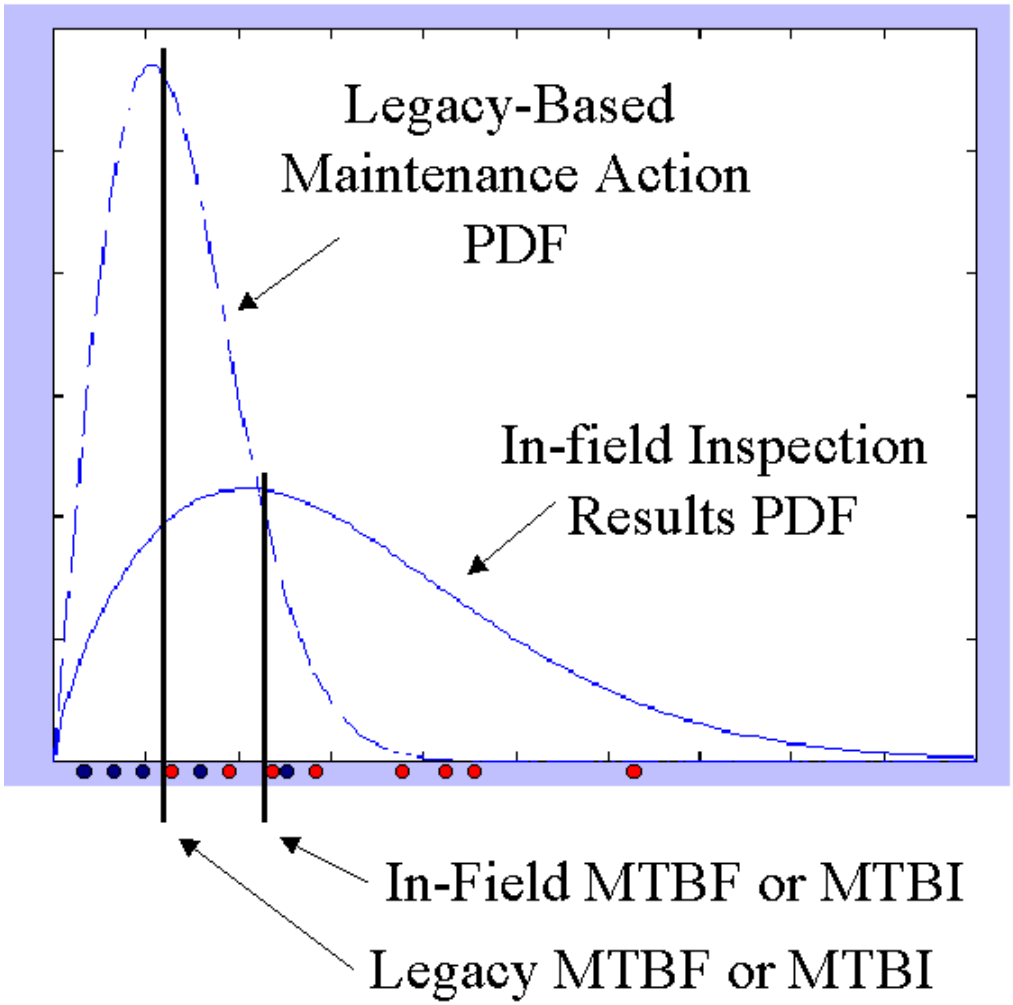
Phase III Follow-On

- Deliver prognostic module tools for use by both the private and public sectors of the industrial base with ultimate integration of CBM prognostic technology into the future TSC HM&E systems.
- Introduced families of prognostic software modules in the FY05 to FY06 timeframe.

Prognostic Module Overview



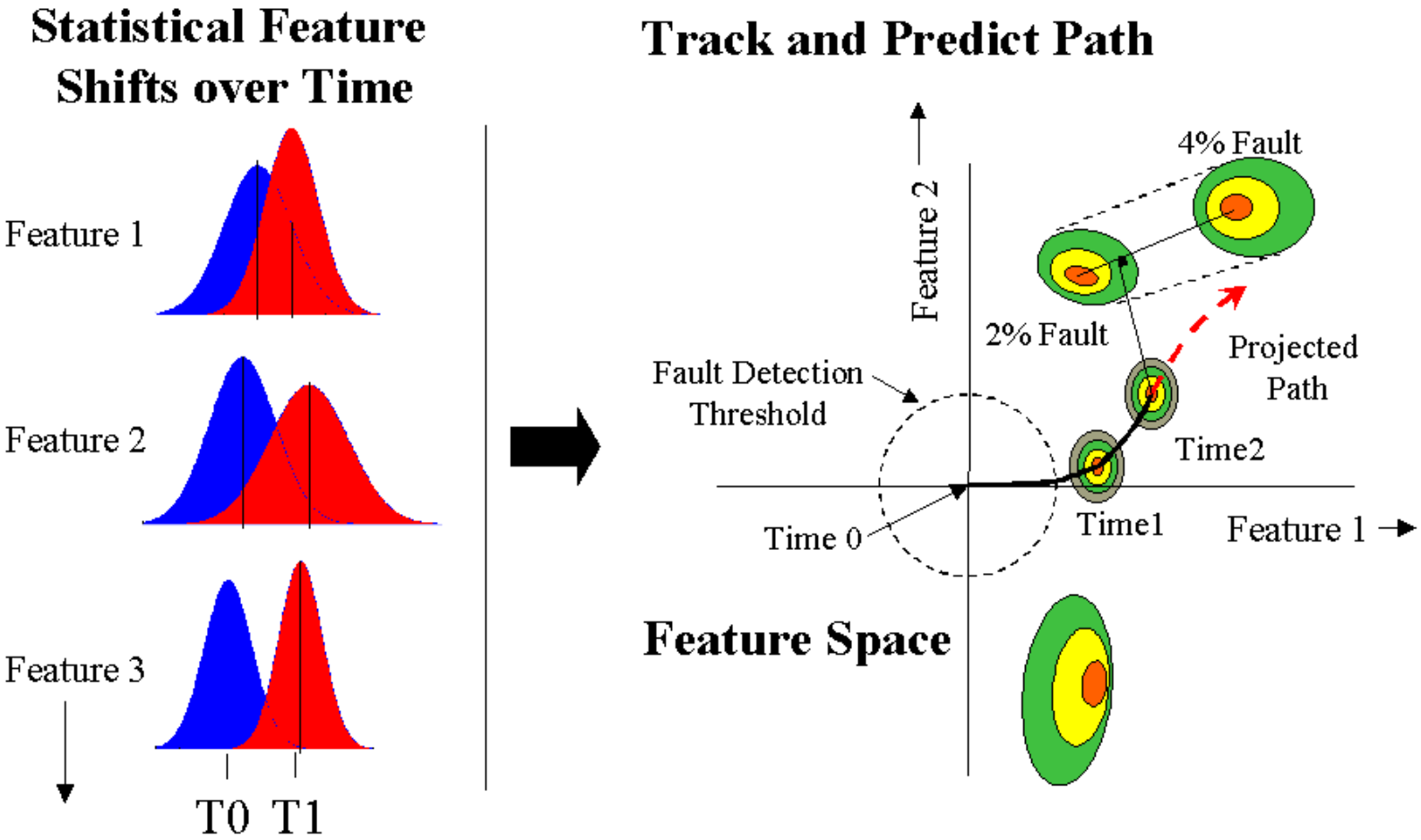
Experience or Historical-based Predictions



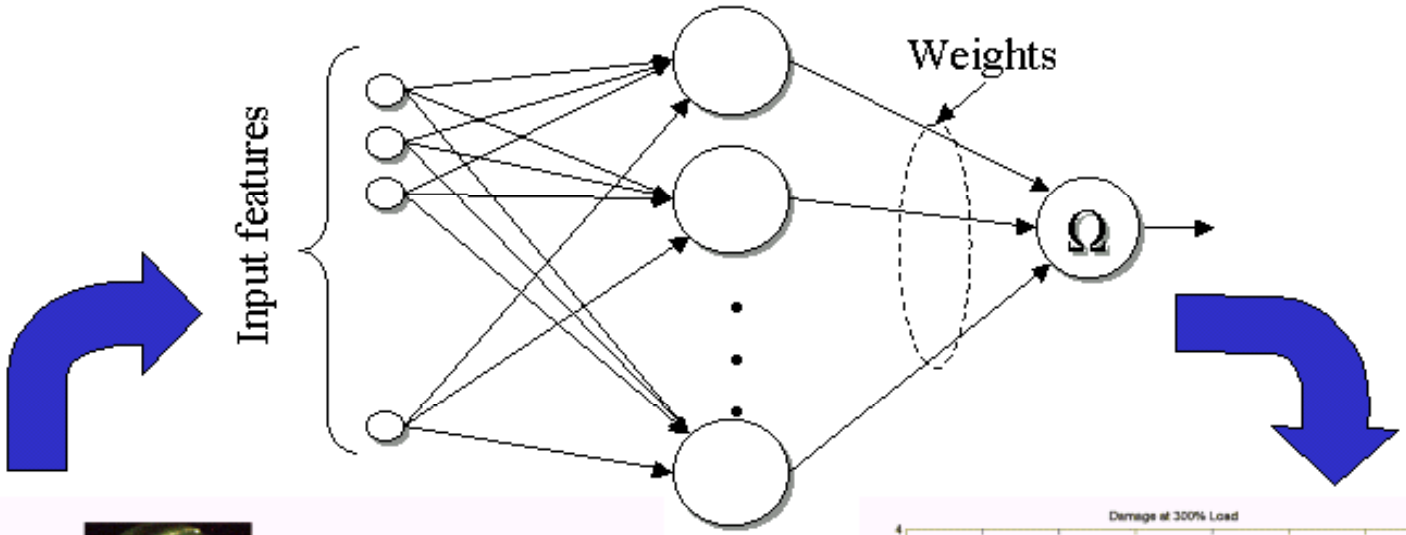
- Weibull Formulation
- Update Capability

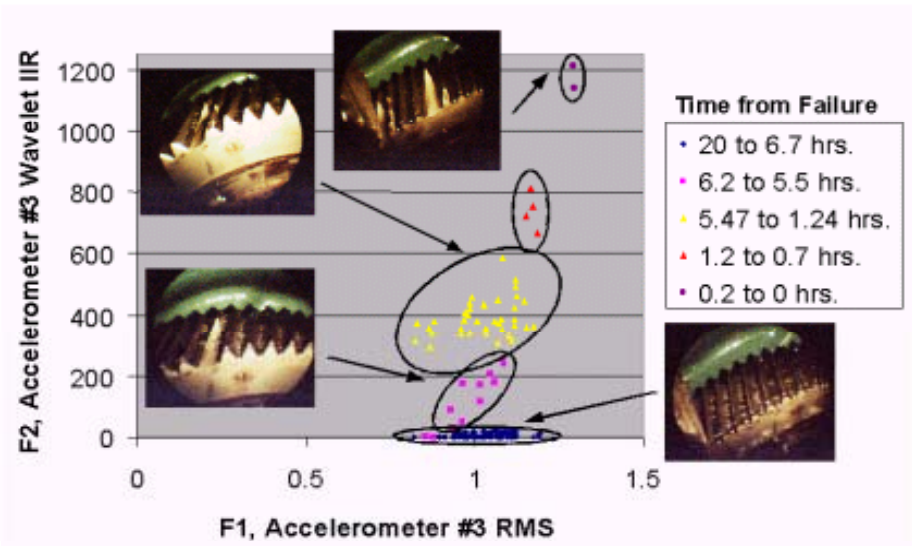
- New Data
- Legacy Data

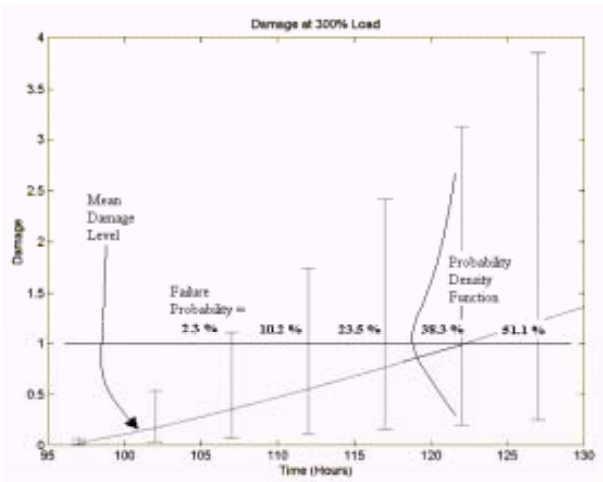
Evolutionary/Trending Prognostics



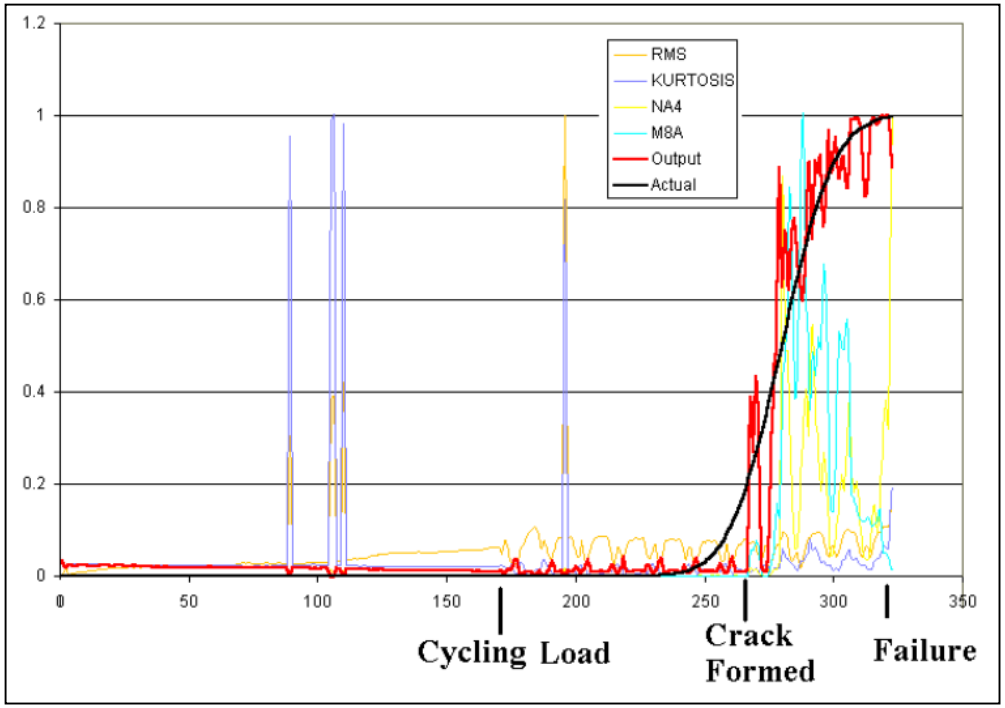
Feature and AI-based Prognostics

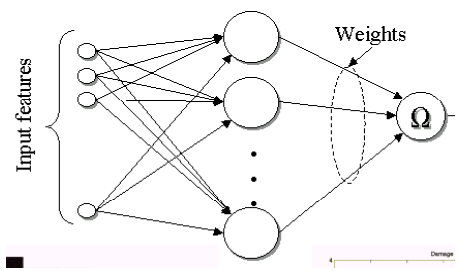






Feature-Based Prognostics





- Multiple features available to detect and diagnose failure modes in mechanical systems
- Fusion of features produces improved fault detection and damage assessment with reduced false alarms
- Neural network trained to associate the fused output with fault progression
- Automated algorithms predict damage onset and progression

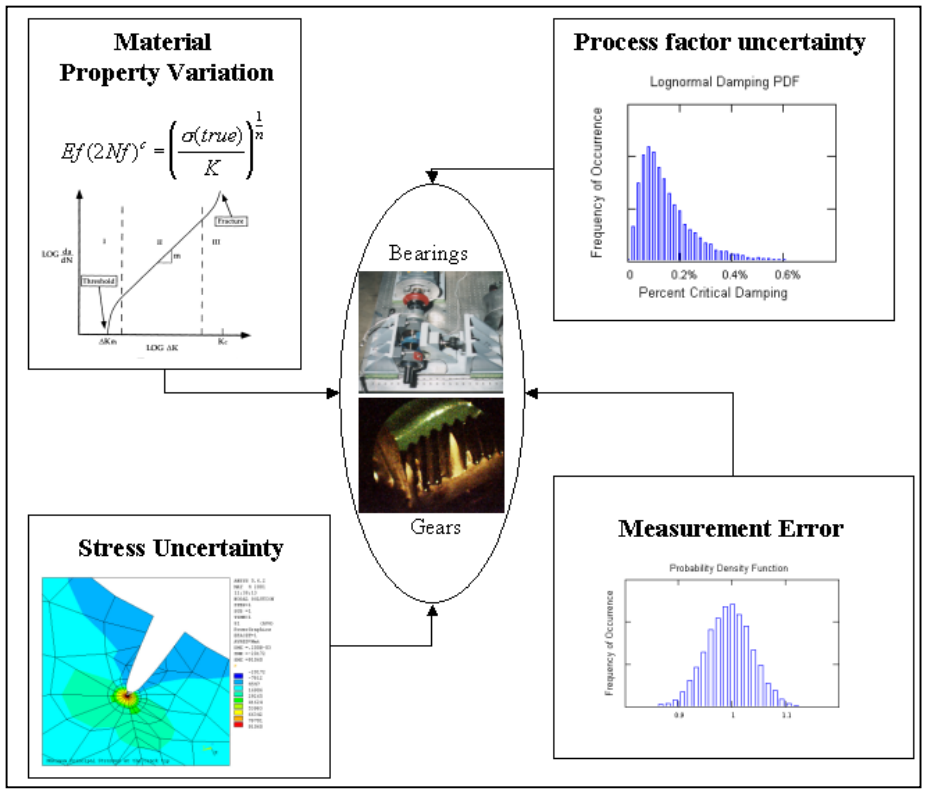
Physics-based Prognostics

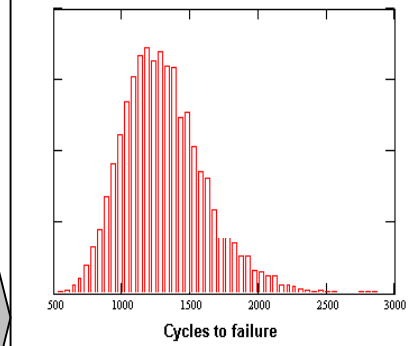
Diagnostic Results

Expected Future Conditions (based on historical conditions)

Experience - Based Information

Failure Mode-Based Stochastic Model





Current & Future Failure Prediction

Phase II Objectives, Workplan and Schedule

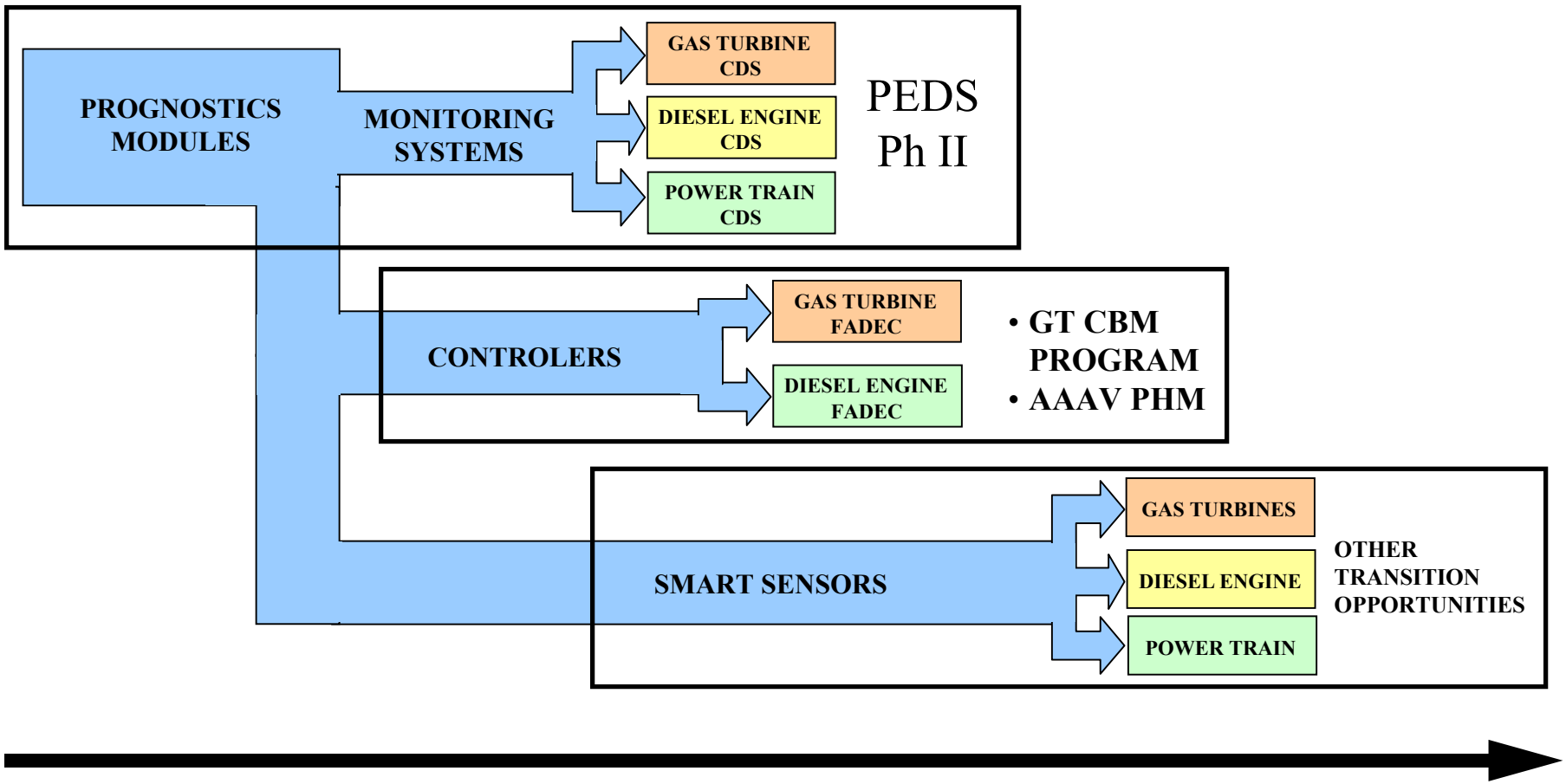
Phase II Objectives

- Demonstrate Architectures and Specifications for Prognostic Enhancements to Diagnostic Systems
- Demonstrate Prognostic Modules in Gas Turbine, Power Systems and Diesel Applications
  - Implement Phase I gear module in PEDS architecture using OSA data flows with XML schema and simple web interfaces.
  - Leverage and integrate existing GT-CBM Waterwash Module – (focused specifically on ICAS integration)
  - Develop Gas Turbine “Hot Section” Component Prognostic Module with advanced lifing/thermo models
  - Perform specific failure mode tests and produce prototype diesel engine prognostic module – (fuel pump, compression loss, etc.)
  - Utilize testbed and field data for validation & calibration
  - Produce appropriate HSI modules for demonstration

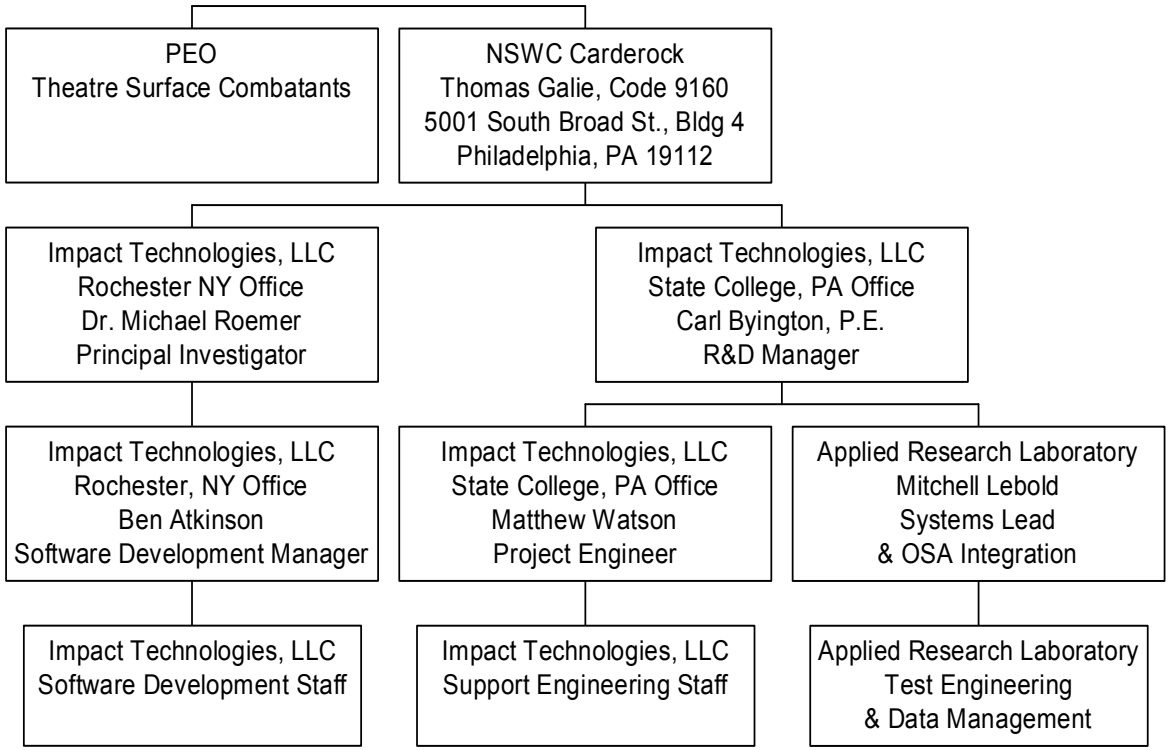
Summary of Phase II Plans

- Prognostic Module Development for Specific PEO-TSC Applications.
  - Feature-Based and Model-Based Integrated Architecture
  - Prognostic Module Tools and Processes
  - Specific Testing for Diesel Prognostics Development
  - Implementing using OSA-CBM Functional Hierarchy
- Identify and Integrate into Selected Naval Ship Application(s).
  - Customize Methodology for Specific Application Integration
  - Integration as ICAS Module or Other Specified Platform
  - Utilize Test and Field Data for Validation & Calibration
  - Build upon existing or use Separate Web-based HSI Concept
- Document Methods, Processes, Lessons Learned for Enhancing Existing Navy Diagnostic Systems

PEDS Timeline



Project Organization



Task 1 - Develop / Mature Generic Prognostic Modeling Approach

Task 2 - Develop Application Specific Prognostic Algorithms

Task 3 - Integrate Data/ Knowledge Fusion Strategies

Task 4 - Prognostic Module Coding and Applications Development

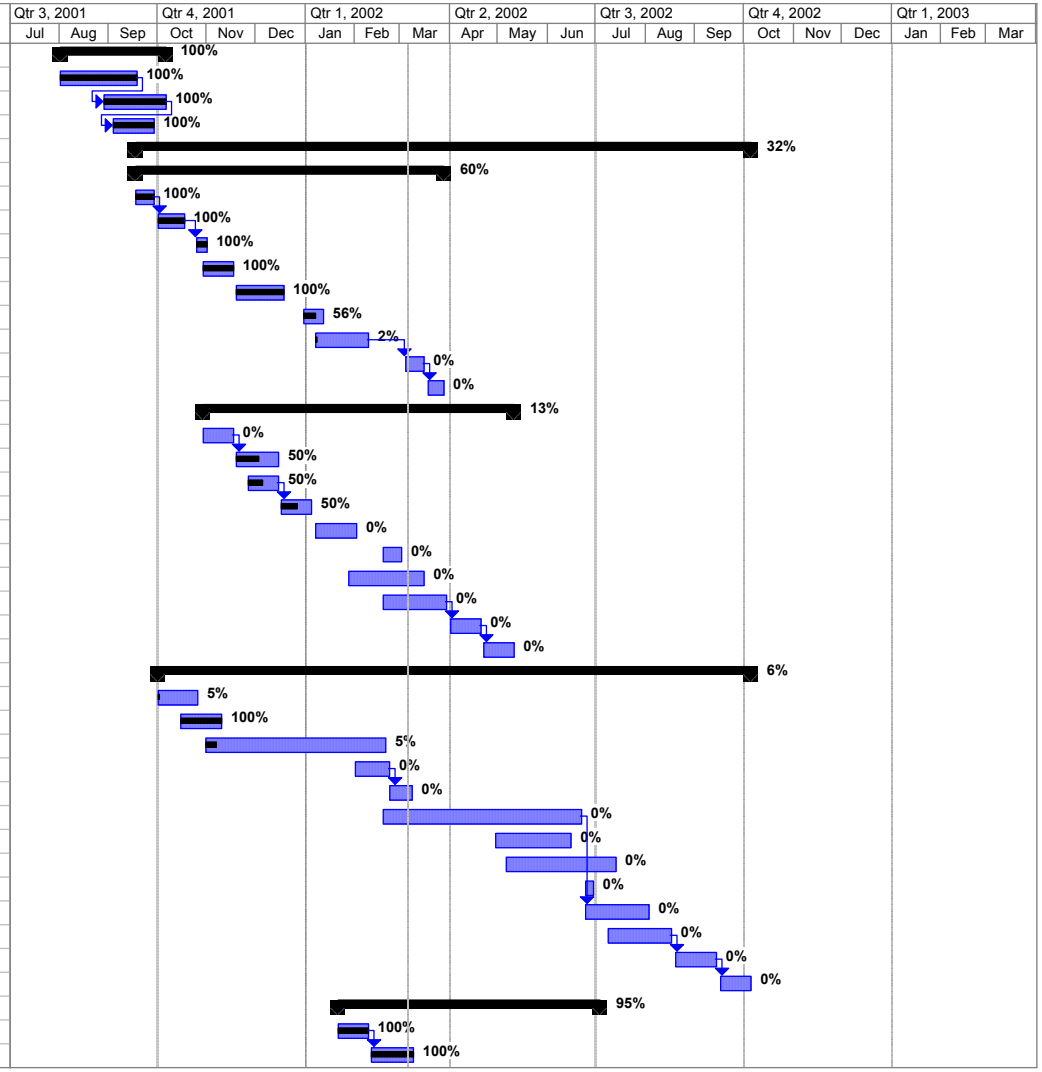
Task 5 - Testing and Verification of Prognostic Modules

Task 6 - Implementation in NSC/LBES shipboard application

Task 7 - Complete HSI development and OSA Tool

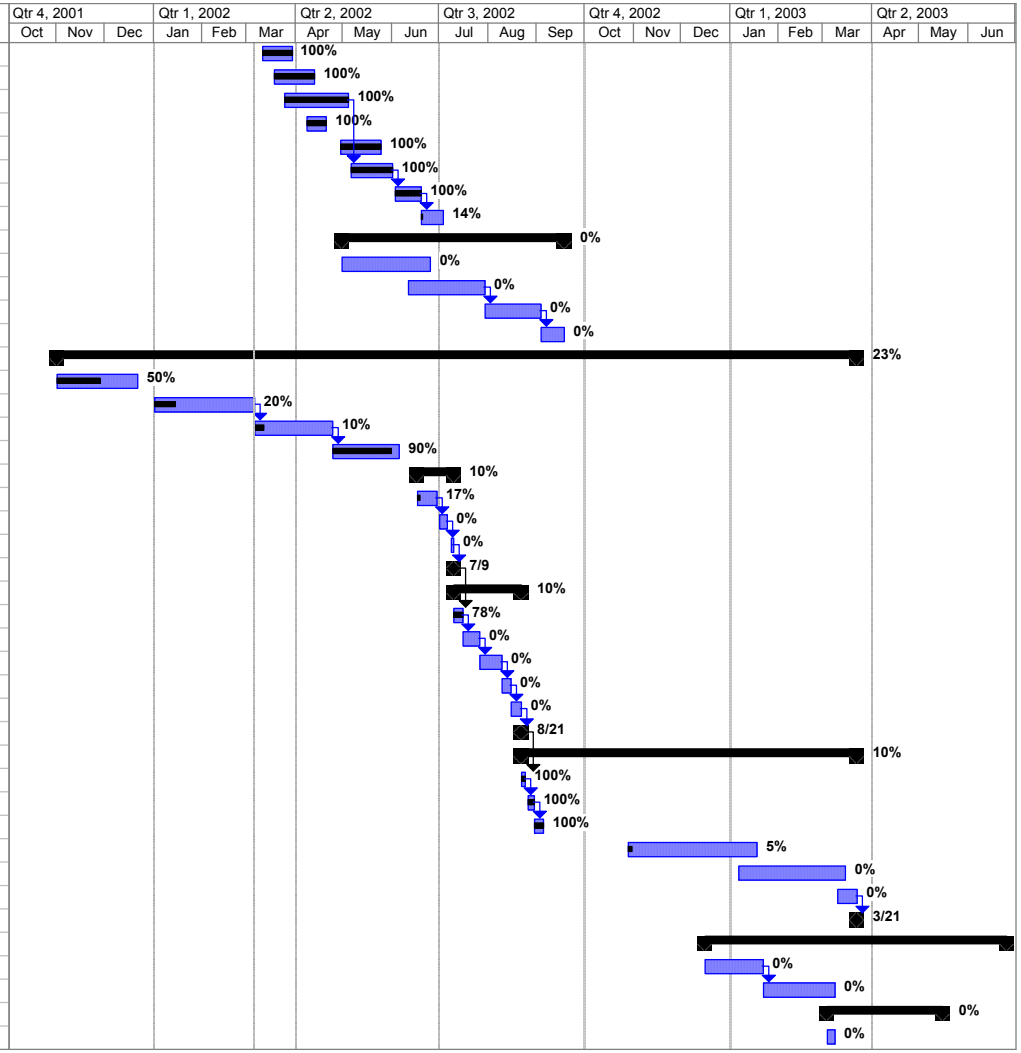
Detailed Program Plan (1)

ID	WBS	Task Name
1	**1**	**Task 1 - Develop / Mature Generic Prognostic Modeling Approach**
2	1.1	Investigate methods of ground truth and information needs
3	1.2	Evaluate integration of experience-base, evolutionary, and model-based predictions
4	1.3	Develop an overall processing concept for prognostics developments
5	**2**	**Task 2 - Develop Application Specific Prognostic Algorithms**
6	**2.1**	**Develop Gas Turbine Performance Prognostic Algorithms**
7	2.1.1	Organize available data and models and determine prog approach
8	2.1.2	Collect data and Apply appropriate standards for raw data sets
9	2.1.3	Apply sensor validation and integrity checks on data
10	2.1.4	Evaluate candidate prognostic features and trending approaches
11	2.1.5	Develop data processing and degradation models using desktop PC tools
12	2.1.6	Apply appropriate data standards for processed features
13	2.1.7	Develop the automated reasoning to be applied for module
14	2.1.8	Integrate the feature extraction, modeling, and reasoning algorithms
15	2.1.9	Verify the integrated prototype and validate on available data
16	**2.2**	**Develop Gas Turbine Components Prognostic Algorithm**
17	2.2.1	Organize available data and models and determine prog approach
18	2.2.2	Collect data and Apply appropriate standards for raw data sets
19	2.2.3	Apply sensor validation and integrity checks on data
20	2.2.4	Evaluate candidate diagnostic/prognostic features
21	2.2.5	Develop data processing and models using desktop PC tools
22	2.2.6	Apply appropriate data standards for processed data
23	2.2.7	Investigate prognostic models for specific failure modes
24	2.2.8	Develop the automated reasoning to be applied for module
25	2.2.9	Integrate the feature extraction, modeling, and reasoning algorithms
26	2.2.10	Verify the integrated prototype and validate on available data
27	**2.3**	**Develop Diesel Engine Component/Performance Prognostic Algorithms**
28	2.3.1	Organize available data and models and determine prog approach
29	2.3.2	Identify critical failure modes for Prognostic modeling
30	2.3.3	Obtain/Generate seeded fault tests
31	2.3.4	Collect data and Apply appropriate standards for raw data sets
32	2.3.5	Apply sensor validation and integrity checks on data
33	2.3.6	Develop and add diagnostic features into a Matlab toolbox
34	2.3.7	Evaluate candidate diagnostic/ prognostic features
35	2.3.8	Develop data processing and models using desktop PC tools
36	2.3.9	Apply appropriate data standards for processed data
37	2.3.10	Investigate prognostic models for specific failure modes
38	2.3.11	Develop the automated reasoning to be applied for module
39	2.3.12	Integrate the feature extraction, modeling, and reasoning algorithms
40	2.3.13	Verify the integrated product and validate on available data
41	**2.4**	**Develop Power Transmission Component Prognostic Algorithms**
42	2.4.1	Organize available data and models and determine prog approach
43	2.4.2	Collect data and Apply appropriate standards for raw data sets



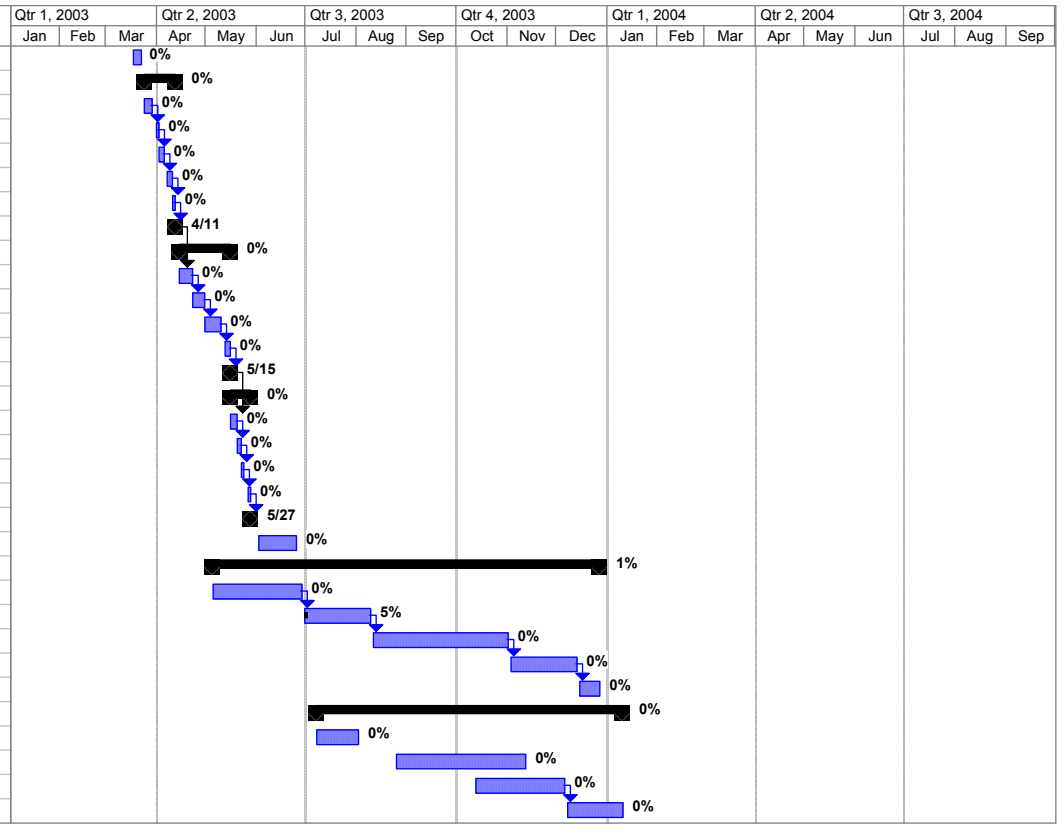
Detailed Program Plan (2)

ID	WBS	Task Name
44	2.4.3	Apply sensor validation and integrity checks on data
45	2.4.4	Evaluate candidate diagnostic/ prognostic features
46	2.4.5	Develop data processing and models using desktop PC tools
47	2.4.6	Apply appropriate data standards for processed data
48	2.4.7	Investigate prognostic models for specific failure modes
49	2.4.8	Develop the automated reasoning to be applied for module
50	2.4.9	Integrate the feature extraction, modeling, and reasoning algorithms
51	2.4.10	Verify the integrated product and validate on available data
52	**3**	**Task 3 - Integrate Data/Knowledge Fusion Strategies**
53	3.1	Evaluate incorporation of historical information and statistics
54	3.2	Employ Bayesian and DS algorithms for updating
55	3.3	Code fusion algorithms in desktop PC tools
56	3.4	Embed in Prognostics Modules
57	**4**	**Task 4 - Prognostic Module Coding and Applications Development**
58	4.1	Identify OSA standard - MIMOSA
59	4.2	Develop and OSA compliant data format and screening tool
60	4.3	Develop an interfacing and Plug-in approach for ICAS
61	4.4	Design and Develop a Plug 'n Play Architecture
62	**4.5**	**Analysis/Software Requirements**
63	4.5.1	Draft preliminary software specifications
64	4.5.2	Incorporate feedback on softwarespecifications
65	4.5.3	Develop delivery timeline
66	4.5.4	Softward Requirements Complete
67	**4.6**	**Design Specifications**
68	4.6.1	Review preliminary software specifications
69	4.6.2	Develop functional specifications
70	4.6.3	Develop prototype based on functional specifications
71	4.6.4	Review functional specifications
72	4.6.5	Incorporate feedback into functional specifications
73	4.6.6	Design complete
74	**4.7**	**Module Development**
75	4.7.1	Review functional specifications
76	4.7.2	Indentify modular/tiered design parameters
77	4.7.3	Assign development staff
78	4.7.4	Develop code in embedded OSA compliant form
79	4.7.5	Incorporate API and appropriate wrappers
80	4.7.6	Developer testing (primary debugging)
81	4.7.7	Development complete
82	**5**	**Task 5 - Testing and Verification of Prognostic Modules**
83	5.1	Develop use cases for planned implementation platforms
84	5.2	Verify operation of prognostic modules with planned interfaces
85	**5.3**	**Software Testing**
86	5.3.1	Develop unit test plans using product specifications



Detailed Program Plan (3)

ID	WBS	Task Name
87	5.3.2	Develop integration test plans using product specifications
88	**5.3.3**	**Unit Testing**
89	5.3.3.1	Review modular code
90	5.3.3.2	Test component modules to product specifications
91	5.3.3.3	Indetify anomalies to product specifications
92	5.3.3.4	Modify Code
93	5.3.3.5	Re-test modified code
94	5.3.3.6	Unit testing complete
95	**5.3.4**	**Integration testing**
96	5.3.4.1	Test module integration
97	5.3.4.2	Identify anomalies to specifications
98	5.3.4.3	Modify code
99	5.3.4.4	Re-test modified code
100	5.3.4.5	Integration testing complete
101	**5.4**	**Deployment**
102	5.4.1	Determine final deployment strategy
103	5.4.2	Develop deployment methodology
104	5.4.3	Secure deployment resources
105	5.4.4	Deploy software in ICAS and demonstrate
106	5.4.5	Deployment complete
107	5.5	Post Implementation Review & Lessons Learned
108	**6**	**Task 6 - Implementation in NSC/LBES shipboard application**
109	6.1	Identify implementation options for shipboard systems
110	6.2	Evaluate implementation within ICAS, FADC (option)
111	6.3	Tailor modules for planned implementation (option)
112	6.4	Meet interface requirements and output HSI structure (option)
113	6.5	Demonstrate implementation (option)
114	**7**	**Task 7 - Complete HSI development and OSA tool**
115	7.1	Develop HSI to demonstrate prognostics module
116	7.2	Design and develop user interfaces for ICAS demonstration (option)
117	7.3	Develop and demonstrate an OSA compiancy tool (option)
118	7.4	Review OSA Compliance of modules and release prototype tool (option)



WBS 1 - Prognostic Architecture Development

- Develop Module Construction and Software Architecture to enable prognostic enhancements using open systems approach
- Accommodate a range of different prognostic approaches and machinery applications
- Exercise architecture over range of prognostic modules types/applications to verify functionality
- Demonstrate in existing diagnostic systems
- Provide guidance for future diagnostic and prognostic system developments to Navy

WBS 2.1 - Gas Turbine Performance Prognostics Module

- Compressor Water Wash Algorithms originally developed for LM2500 and Allison 501-K34 GT Engines on USS Laboon (NSWCCD-SSES Code 9334)
- Predict compressor performance degradation rates due to salt deposit ingestion using LBES data sets
- Outputs optimum time for on-line water washing or crank washing from a cost/benefit standpoint
- Combines Evolutionary-Based (Probabilistic Analysis) and Historical/Experience Based (Fouling Test Results) approaches using advanced forecasting methods
- Implementation candidate for ICAS deployment
  - Written in C++ Compiled as dll using DDI
  - Pseudo-sensor output of module

Dll – Dynamic Link Library **DDI- Demand Data Interface**

WBS 2.2 - Gas Turbine Component Prognostics Module

- Investigating feasibility of turbine section blade module for LM2500 with LCM
- May require reverse engineering blade geometry to produce FE model
- Utilize stress and life usage prediction methods
- Couple with performance model for thermomechanical fatigue prognostics
- Transition through ICAS and Gas Turbine CBM program

LCM – Life Cycle Manager

WBS 2.3 - Diesel Engine Prognostics Module

- Collect steady state performance and transient event data on laboratory engine (DEMDTB)
- Identify specific low bandwidth features for performance fault classification and trending
- Develop Evolutionary Prognostics approach based upon statistical trackers
- Identify high bandwidth features to track clogged injectors, fuel pump failure, and cylinder problems
- Develop State Space and AI based prognostics modules
- Potential implementation within ICAS or diesel engine controller
  - Written in C++ compiled as shared library

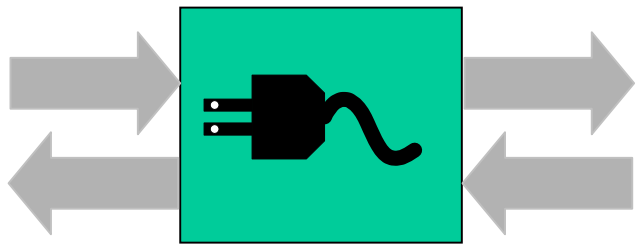
WBS 2.4 - Power Transmission Component Prognostics Module

- Extension of Gear Prognostic Module developed in Phase I to include OSA-CBM constructs and reorganized into Prognostics Architecture
- Predicts MTTF using a combination of Model-Based (Physics) and Feature-Based Prognostic approaches (Dempster Shafer Fusion)
- Outputs MTTF and Probability of Failure with Confidence Bounds
- Implemented using Java Native Interface (JNI) language
  - Extension of Java (Sun) allows calls to native (legacy) code (C, C++, etc)
  - Development environment readily supports multiple middleware
- Potential transition through smart sensing developments

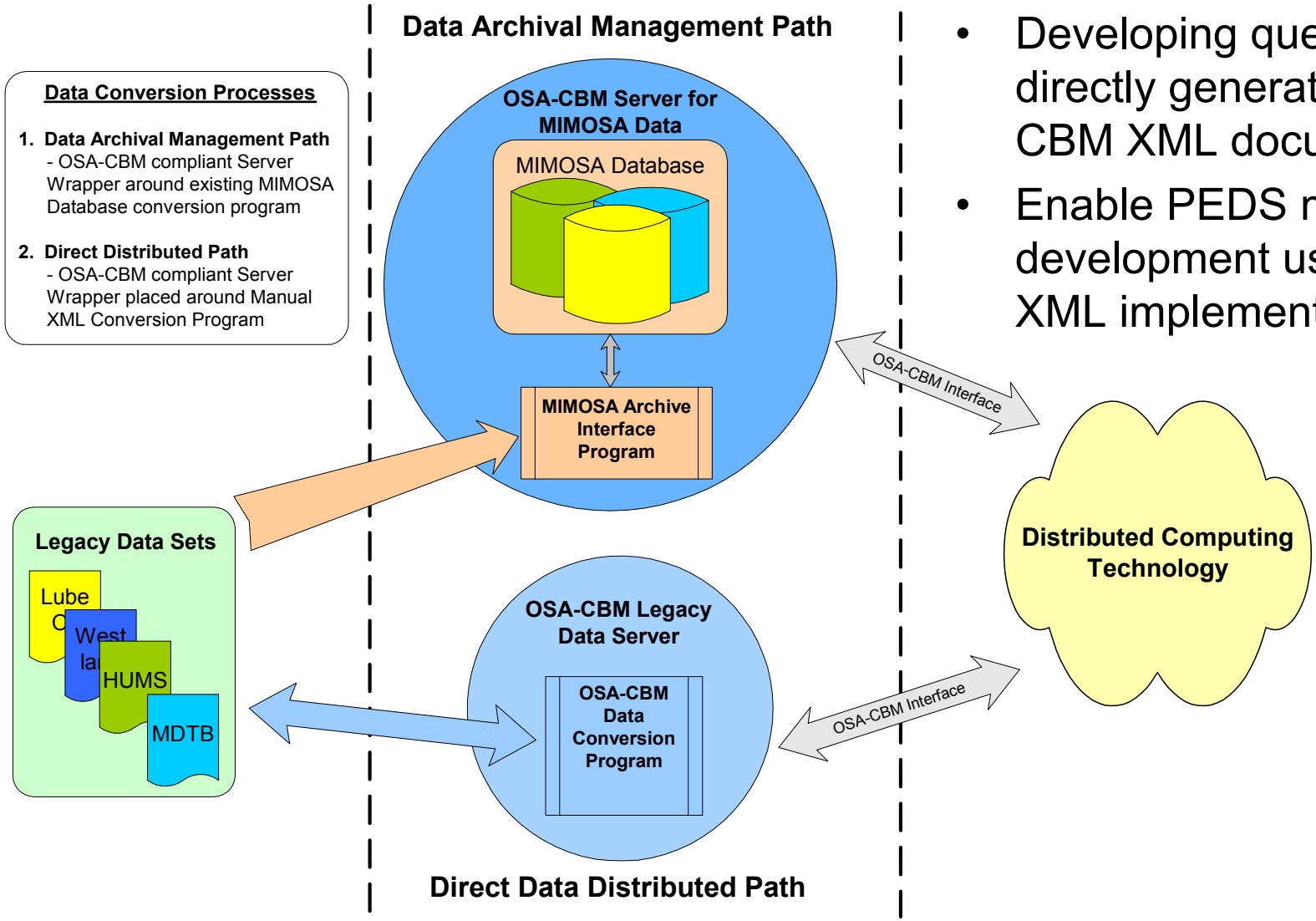
OSA and Prognostics Architecture Development

Open System Architecture Attributes

- Non-propriety standard interface format (ie. A publicly known input and output interface specification)
- Independent sub-module design technology
- Allows any 3rd party to use a system module within their system
- MIMOSA CRIS and OSA-CBM Functional Architectures
- Communications between modules is accomplished using established middleware technologies

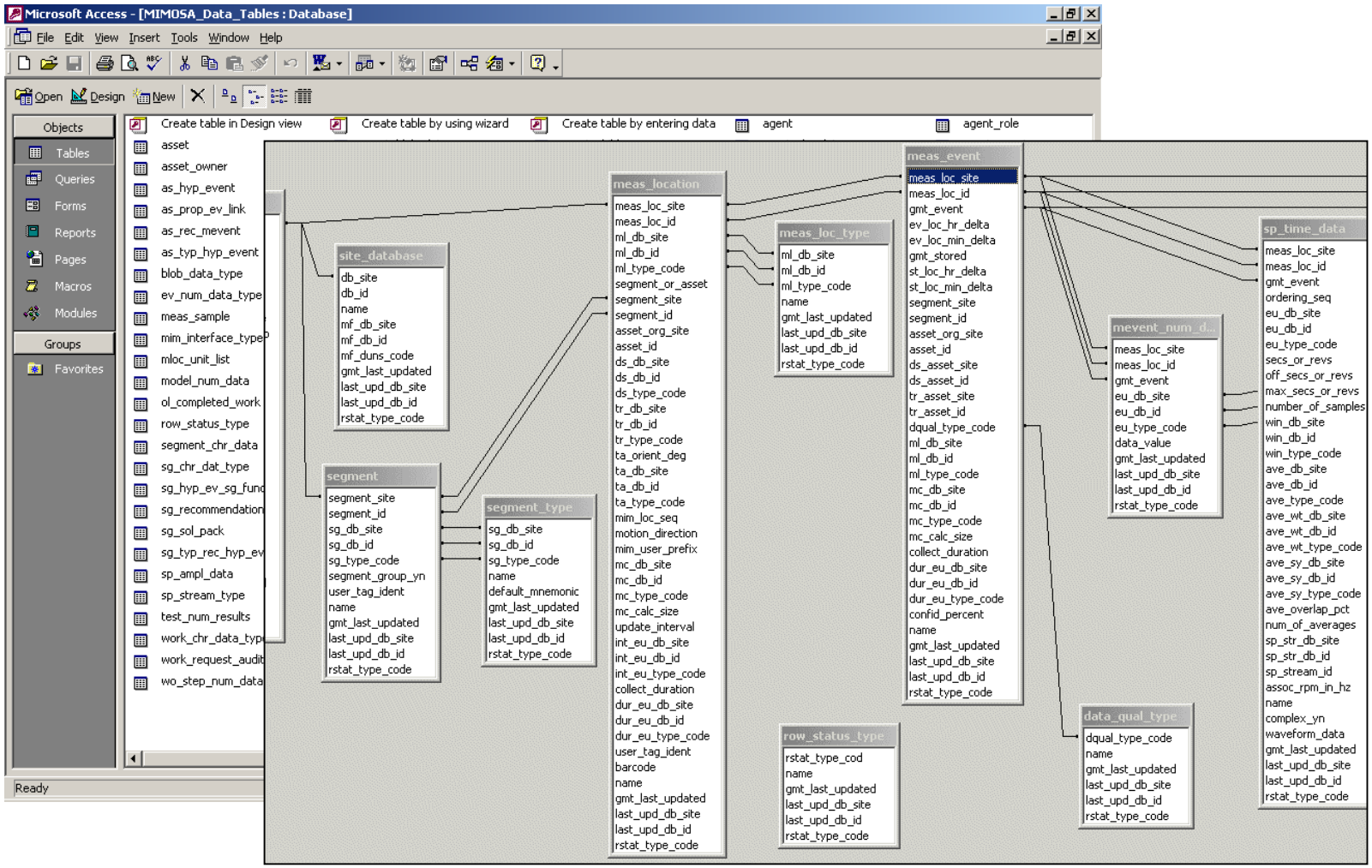


Using MIMOSA Databases for PEDS Developments



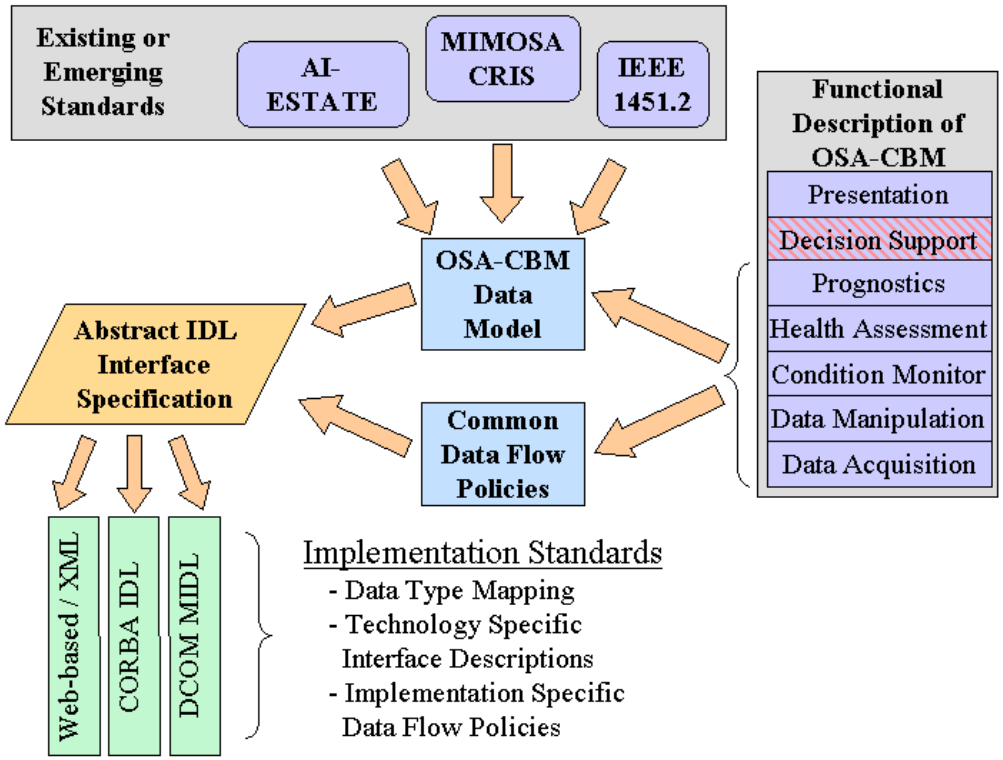
- Developing queries to directly generate OSA-CBM XML documents
- Enable PEDS module development using XML implementations

Microsoft Access Implementation



OSA-CBM Data Models and Flows

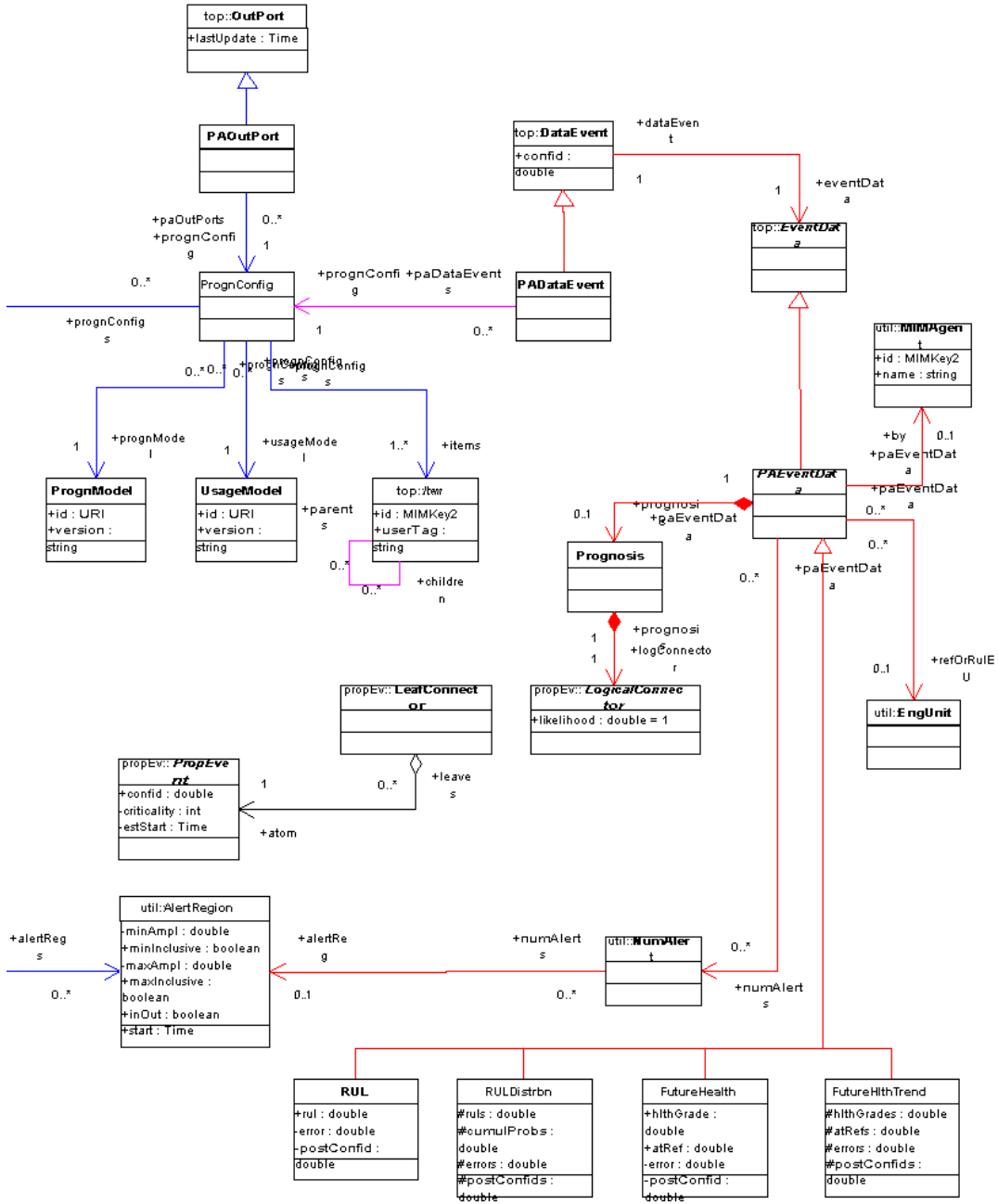
- Object-oriented Data Model defined using UML (Unified Modeling Language)
  - OSA-CBM Release 1.00
- Encapsulation of proprietary algorithms within modules and requirements for external interfaces at each functional level
- Time-based or Event-based push of information
- Association of information consumers with triggered events
- Abstract IDL (Interface Definition Language) describes common structures and interface configurations



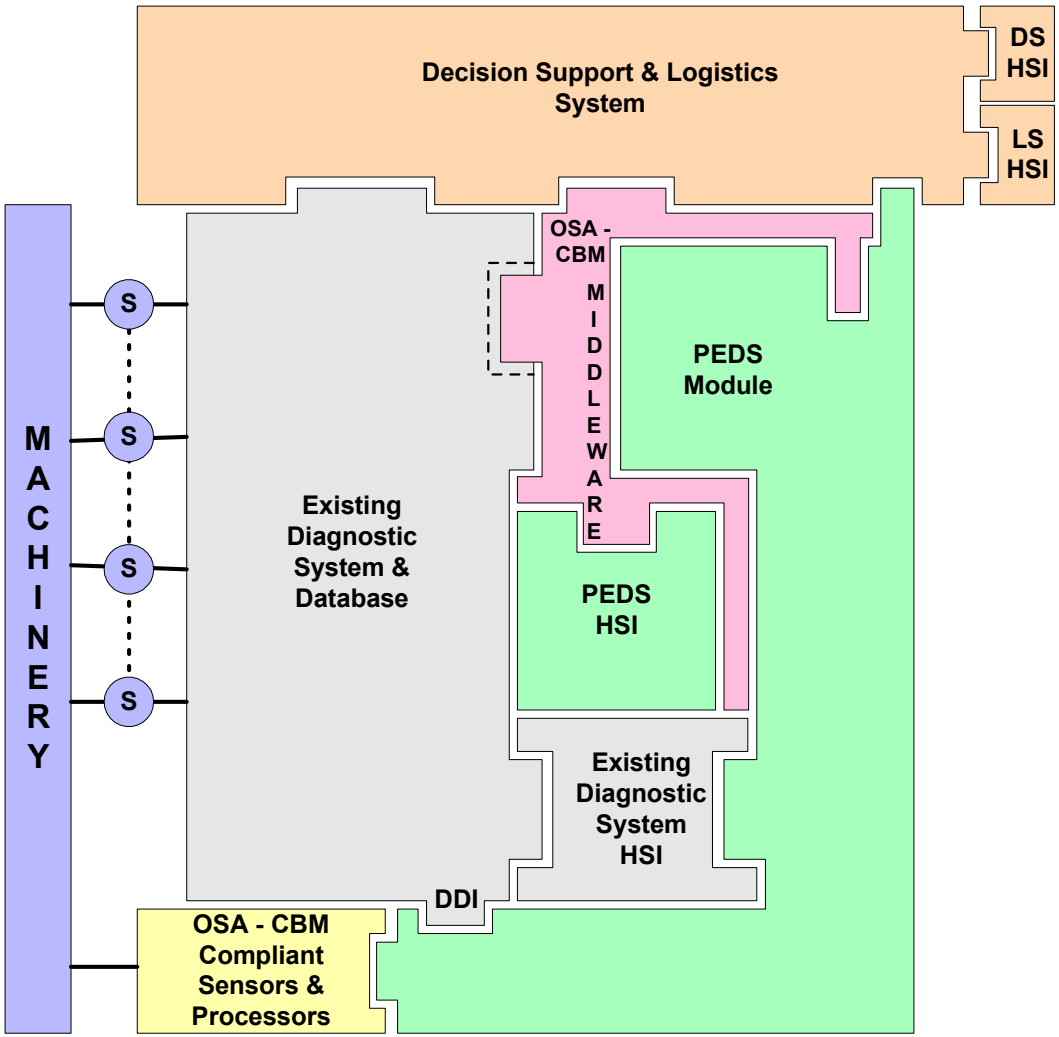
- Bridges are necessary to operate between middleware IDL's

OSA-CBM UML Model of a Prognostic Layer

UML: Unified Modeling Language



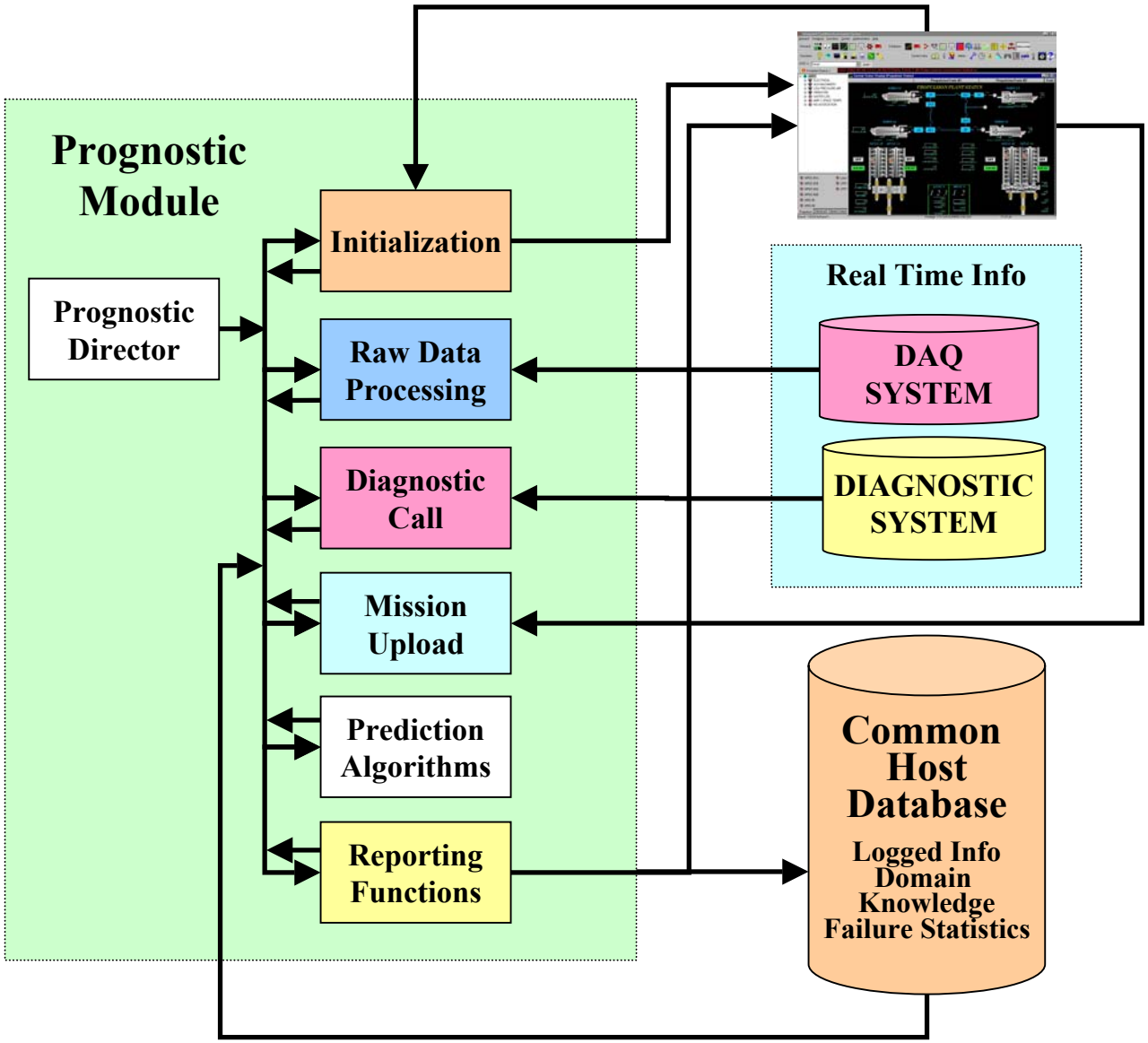
Putting the Pieces Together



- Depicts connections and communications between the existing elements and system enhancements
- Proprietary interfaces or OSA-CBM middleware to “glue and hook” modules together
- HSI module separate in functionality
  - Use existing proprietary interface or OSA compliant HSI module

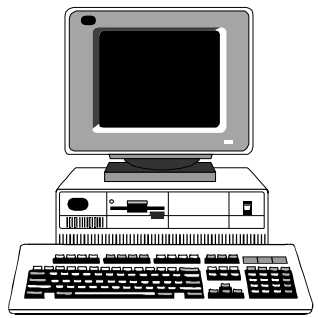
Architecture Development Overview

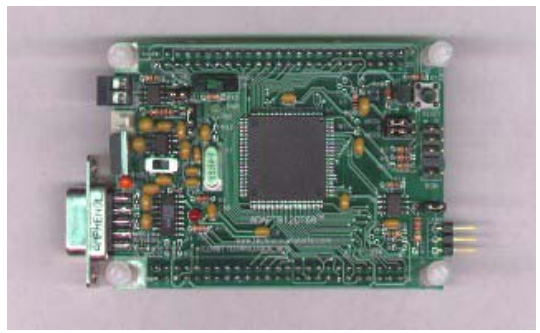
PEDS Architecture and Interfaces



Software Requirements

- Open Systems Environment
- Interface With ICAS
- Modularity/Reusability
- Multi-platform Use
- Operation in Multiple Modes
- Robustness
- Performance
- Logging and I/O





Trade Study: Programming Languages

Design Choices		SUN JAVA (exe)			
	Req. Weight	*PROS*	*CONS*	*Score*	*Pts*
Requirements					
Open Systems Environment The module shall define its interfaces for an open systems environment as defined by MIMOSA, OSA CBM and derivative Standards	20	Supported classes from SUN and 3rd party vendors allows easier interfacing with middleware (CORBA, XML, JAVA-RMI)	Bridging will be needed for DCOM	4	**16**
Interface With ICAS The module developed in this project will interface with ICAS. Therefore it will be written to operate in Windows 98/NT.	15		Unknown at this time.	0	**0**
Modularity/Reusability The CSCI shall be modular to adapt to the end-use constraints for execution speed, response time, and throughput.	12	Because of JAVA's more pure OOP nature, modularity will be high		4	**9.6**
Multiplatform Use The module should be able to run in multiple environments.	15	Java programs are compiled into machine-independent bytecodes; they run consistently on any Java platform	Performance suffers from added portability. Java. Compared to C Java is relatively unproven in embedded systems.	3	**9**

- Comparison of Sun Java, J++, C++, and JNI (executables and shared libraries)
- Pros and Cons defined with respect to PEDS requirements
- Weights assigned for each developed requirement and scores compiled
- Using specific environments/ languages to develop different modules

Middleware Implementation Study

	COM/DCOM/COM+	CORBA	Java Platform	Web/HTTP
Cross-language support	+	+	-	+
Cross-platform support	-	0	0	+
Server Components	+	+	+	+ Web servers
Networking	+ DCOM	+ IIOP	+ RMI/IIOP	+ HTTP
Transactions	+ MTS, DTC	0 Implementation Dependent	0 API only	-
Messaging	+ MSMQ, queued comonents and events in COM+	0 Implementation Dependent	0 API only	-
Naming	0 registry + ADS in COM+	+ available in all implementations	0 JNDI - API and reference implementation	0 DNS only
Product Maturity	+ COM/DCOM 0 COM+	+	0 J2EE	- new technology for distributed systems
Vendor Outlook	+ Microsoft	- some companies abandon the technology	+ Sun	- not clear for distributed systems

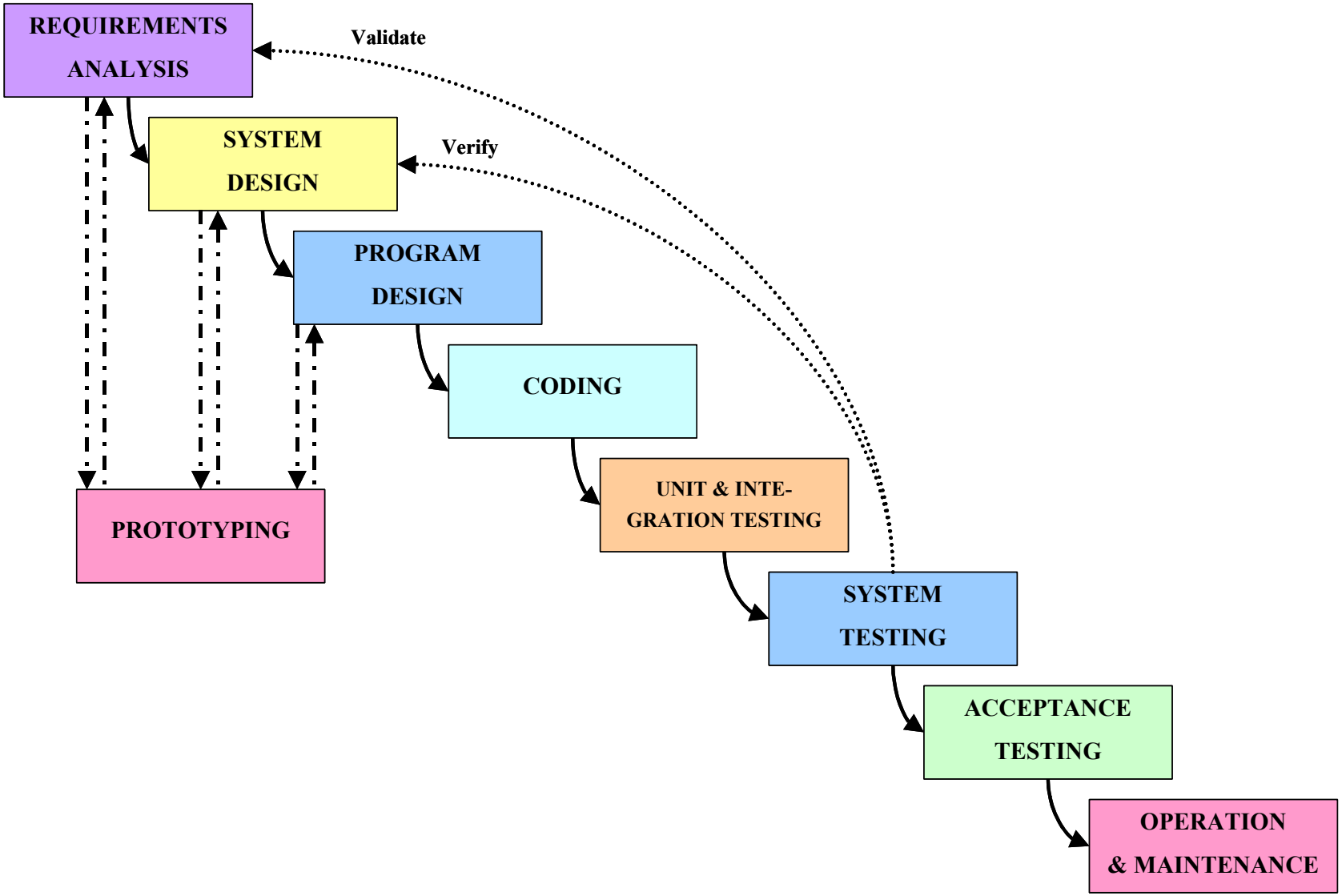
+ Satisfactorily Implemented
- Partially Implemented
0 Not Sufficiently Implemented

Cervinka,O., Bezdicek, J., and Hejda, P., "Middleware for OSA-CBM", Jan 18, 2000

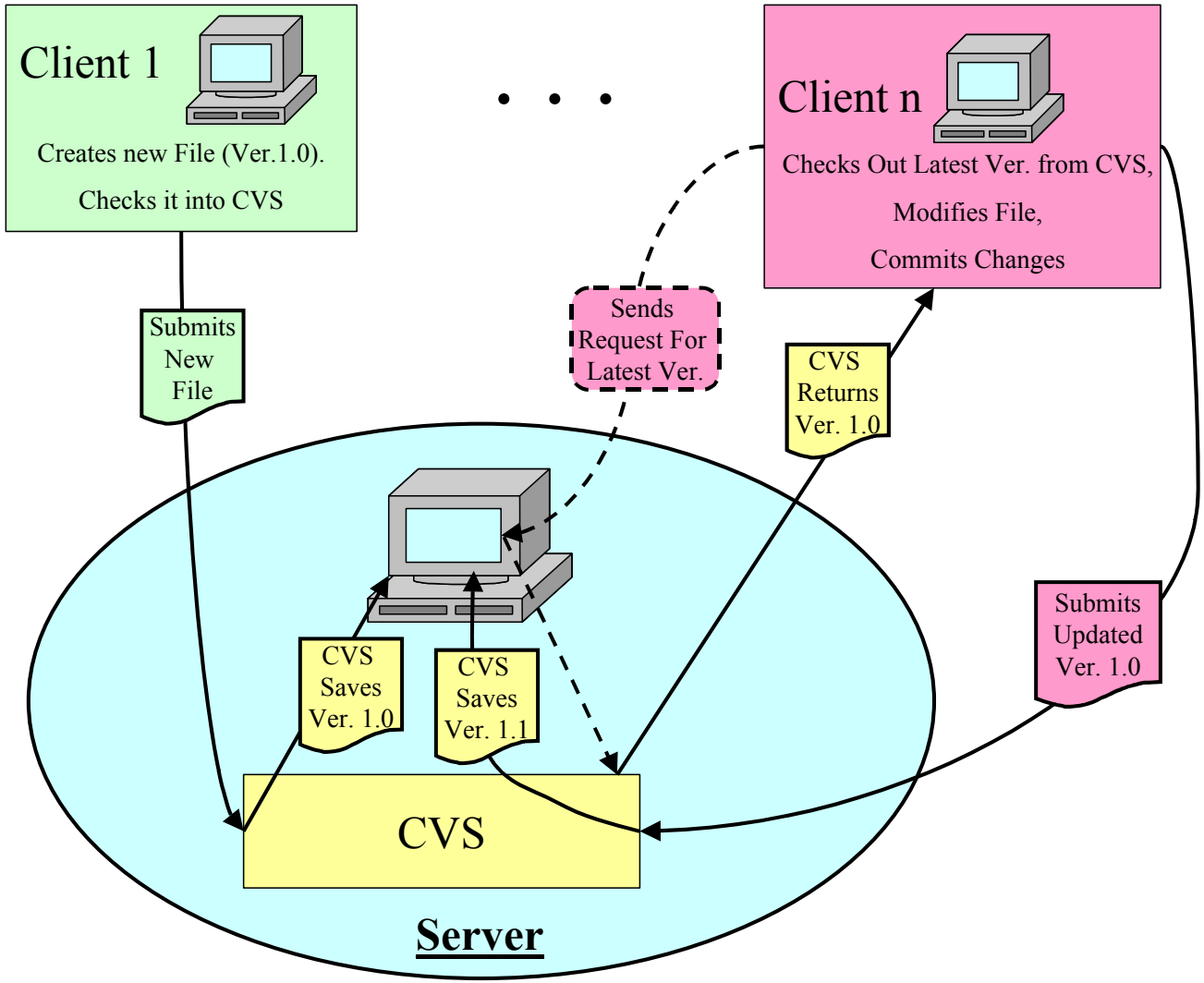
Implementation Approach

- C++ (dlls), Java (exe), and JNI will be used
- C++ (dlls) will be used for modules planned for ICAS integration
- JNI will be used to explore the advantages and disadvantages of this approach
- XML will be used as “middleware” technology
- Select specific bridges to demonstrate
  - Modules will not be expected to interface with every possible middleware

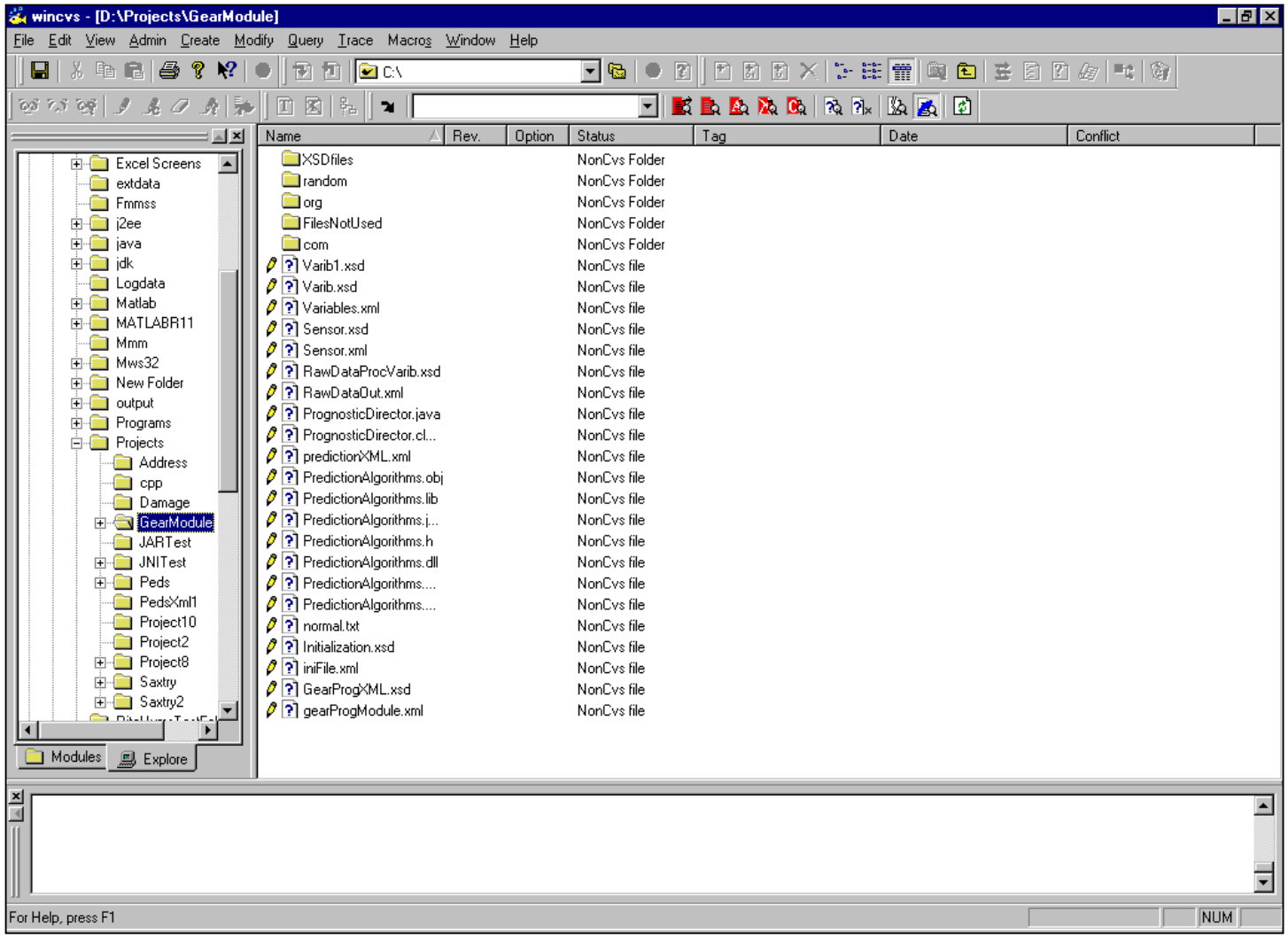
Module Development Process



Document Version Control



WinCVS

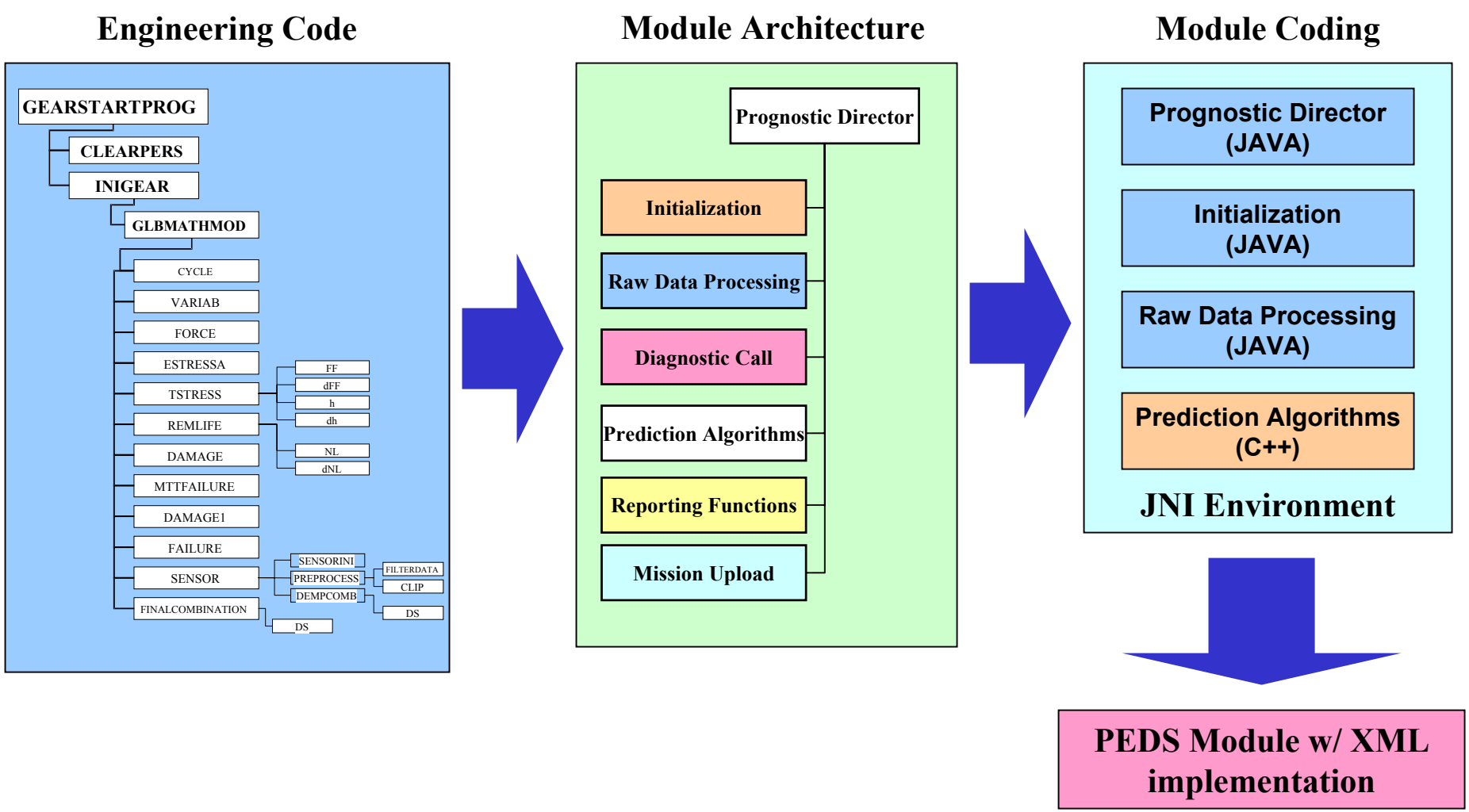

wincvs - [D:\Projects\GearModule]
File Edit View Admin Create Modify Query Trace Macros Window Help
C:\
Name Rev. Option Status Tag Date Conflict
Excel Screens
extdata
Fmmss
j2ee
java
jdk
Logdata
Matlab
MATLABR11
Mmm
Mws32
New Folder
output
Programs
Projects
Address
cpp
Damage
GearModule
JARTest
JNITest
Peds
PedsXml1
Project10
Project2
Project8
Saxtry
Saxtry2
Modules
Explore
XSDfiles NonCvs Folder
random NonCvs Folder
org NonCvs Folder
FilesNotUsed NonCvs Folder
com NonCvs Folder
Varib1.xsd NonCvs file
Varib.xsd NonCvs file
Variables.xml NonCvs file
Sensor.xsd NonCvs file
Sensor.xml NonCvs file
RawDataProcVarib.xsd NonCvs file
RawDataOut.xml NonCvs file
PrognosticDirector.java NonCvs file
PrognosticDirector.cl... NonCvs file
predictionXML.xml NonCvs file
PredictionAlgorithms.obj NonCvs file
PredictionAlgorithms.lib NonCvs file
PredictionAlgorithms.j... NonCvs file
PredictionAlgorithms.h NonCvs file
PredictionAlgorithms.dll NonCvs file
PredictionAlgorithms.... NonCvs file
PredictionAlgorithms.... NonCvs file
normal.txt NonCvs file
Initialization.xsd NonCvs file
iniFile.xml NonCvs file
GearProgXML.xsd NonCvs file
gearProgModule.xml NonCvs file
For Help, press F1
NUM

Specific Module Development Status

Specific Module Development

- Developed PEDS Prognostic Architecture implemented on specific modules
  - Power Transmission Component Failure, Diesel Engine Performance/Component, Gas Turbine Performance, and Gas Turbine Component
- Simple Graphical User Interface module developed to demonstrate outputs
- Modules developed using a number of programming environments
  - C++ dlls, Java (Sun) exe, and JNI exe
- Predominately use XML and maybe CORBA middleware

Translating Into Prognostic Elements (PTC Example)



Power Transmission Component Prognostic Module

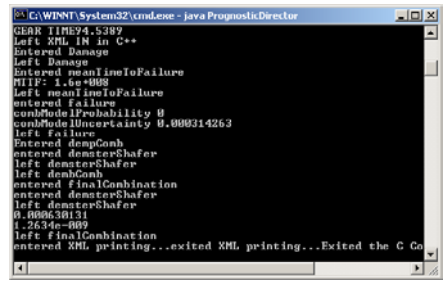
PTC Module Version 1.0.0

Prognostic Director (Java)

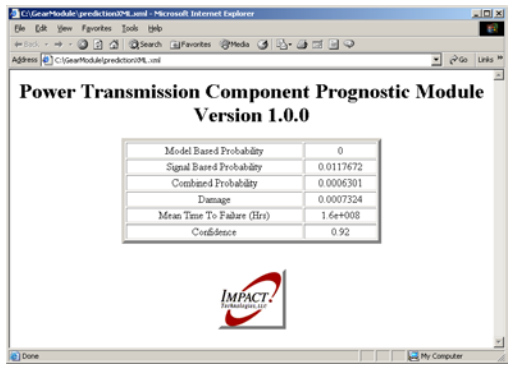
```
import com.impact.RawDataProc.*;
import com.impact.Initialization.*;

public class PrognosticDirector
{
        public static void main(String[] args)
        {
                String dataFileName = null;

                //get data for the first run
                dataFileName = Initialize.initializeData();

                while (dataFileName != null){
                        //output XML
                        dataFileName = Initialize.initializeData();

                        RawDataProcessing.RawData(dataFileName);

                // Call the PredictionAlgorithms class in every case
                        PredictionAlgorithms.CallC();

                // get data

                }
        }

}
```

Prediction Algorithms (C++)

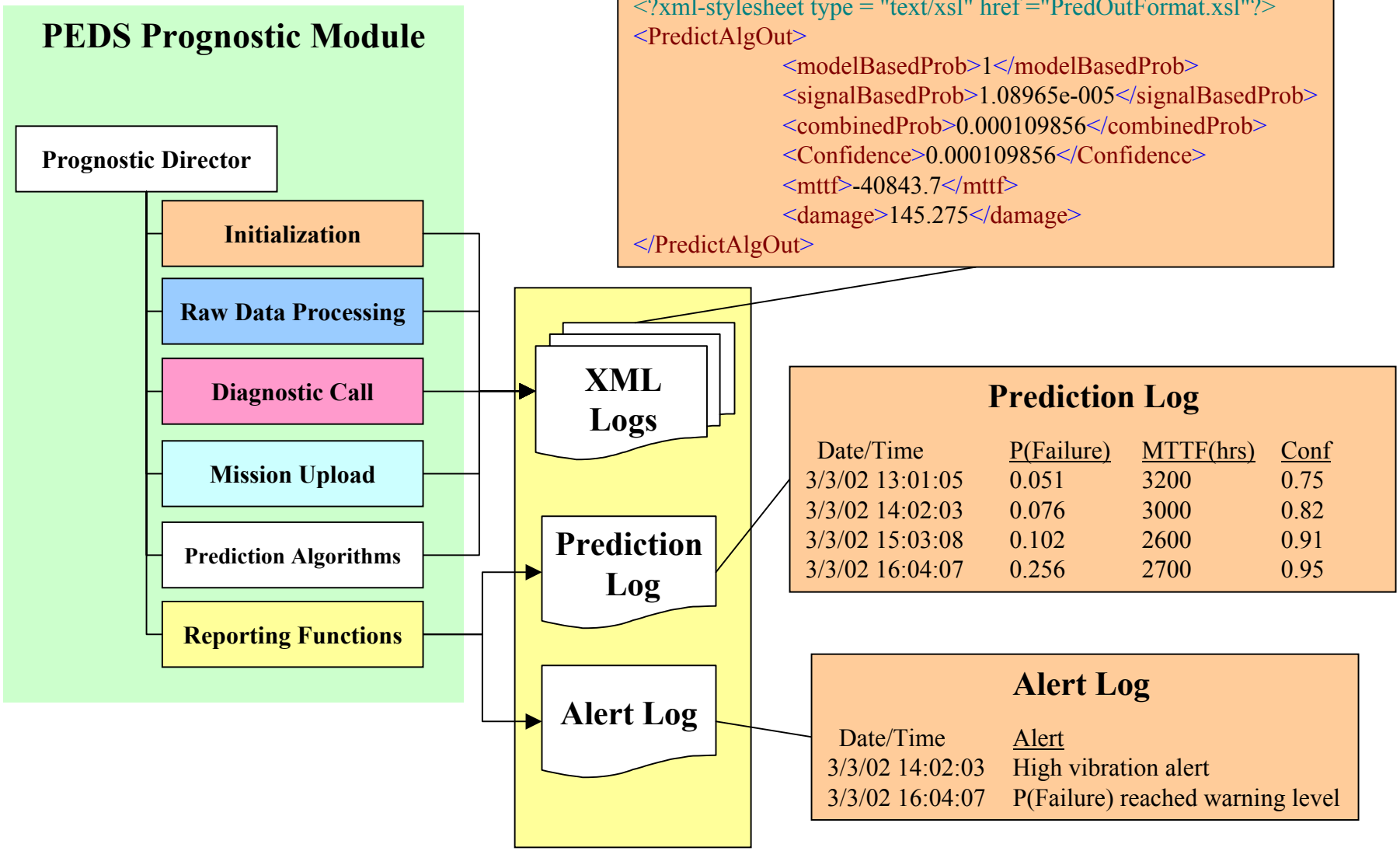
```
//Included for JNI
#include <jni.h>
#include "PredictionAlgorithms.h"
#include <stdio.h>
#include <ctype.h>
```



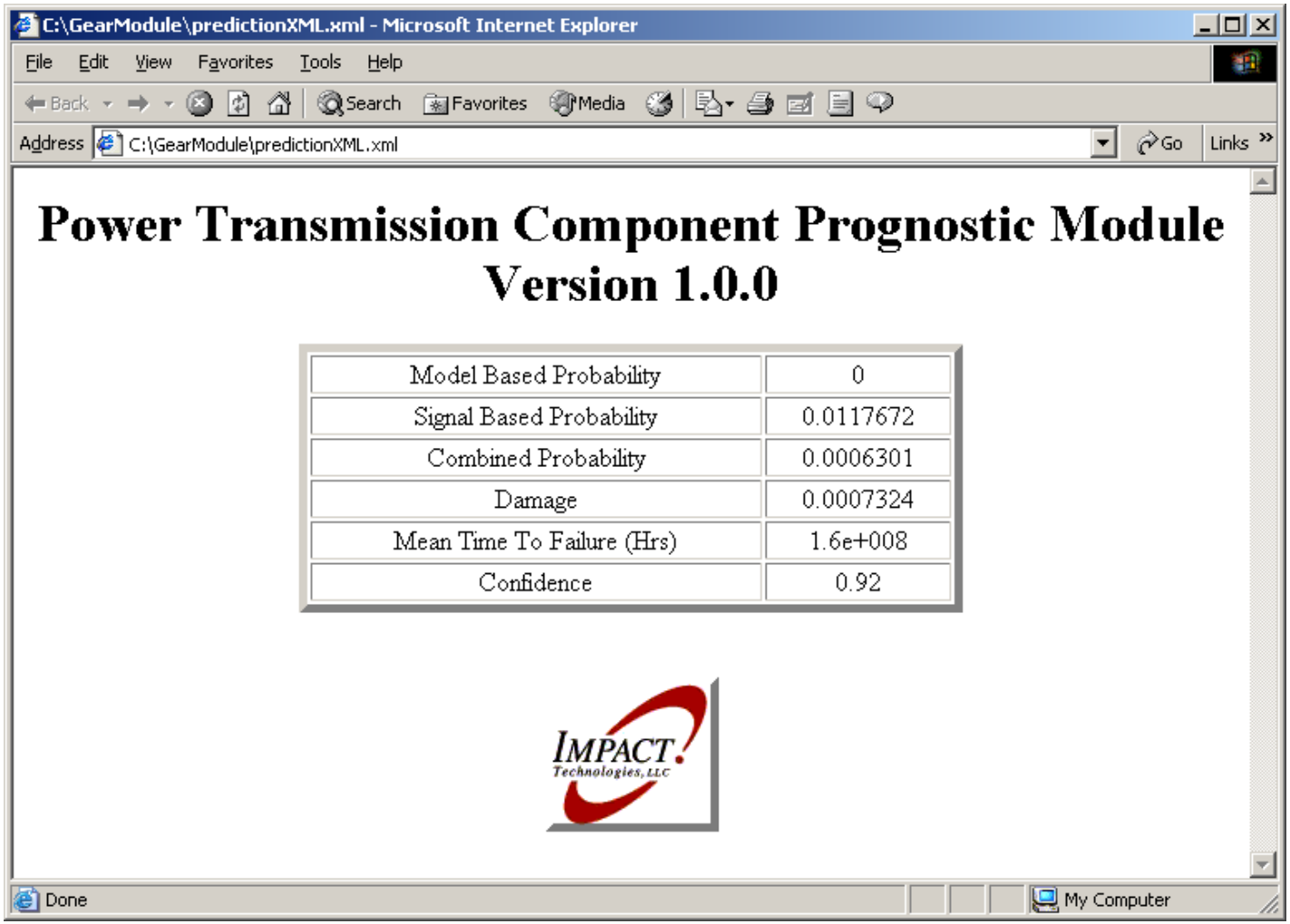
HSI Version 1.0.0



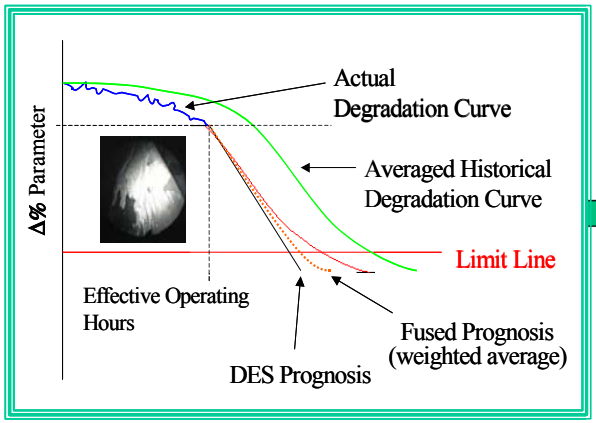
Logging Functionality

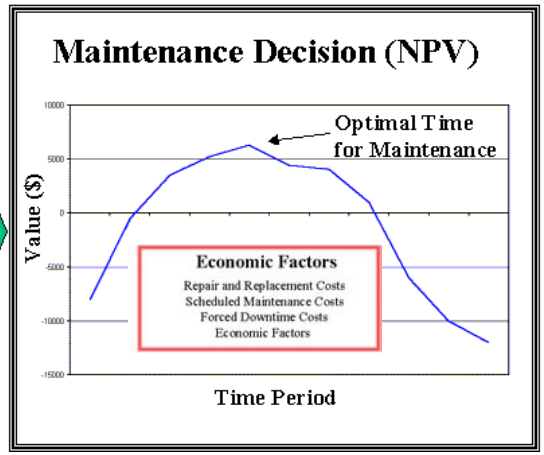


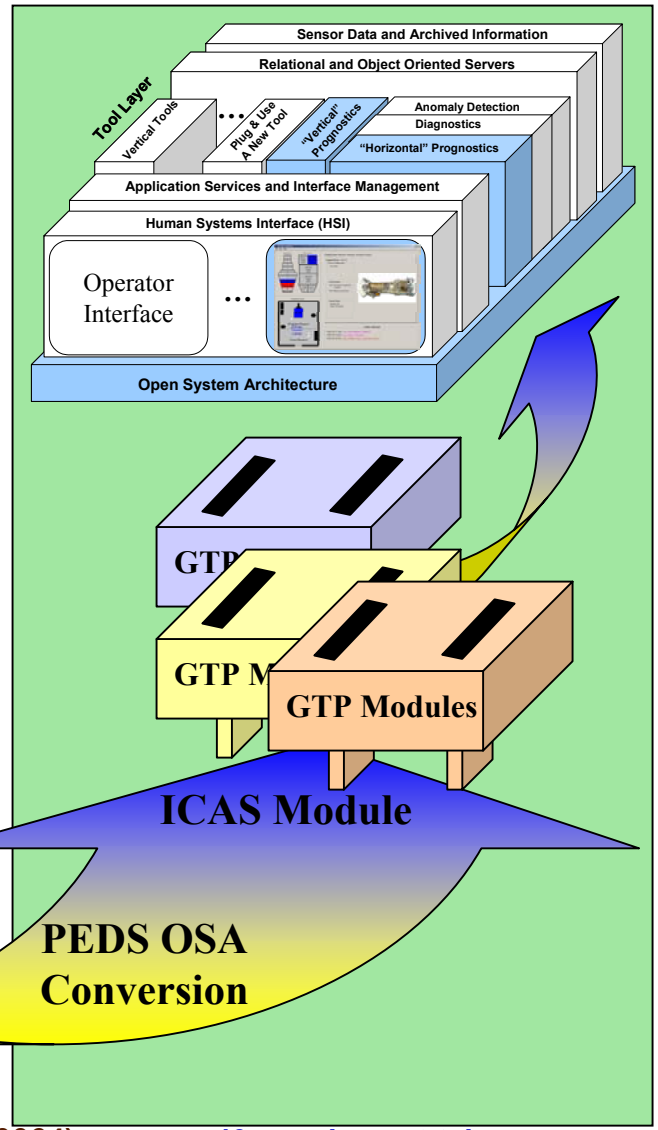
Power Transmission Component Prognostic Module



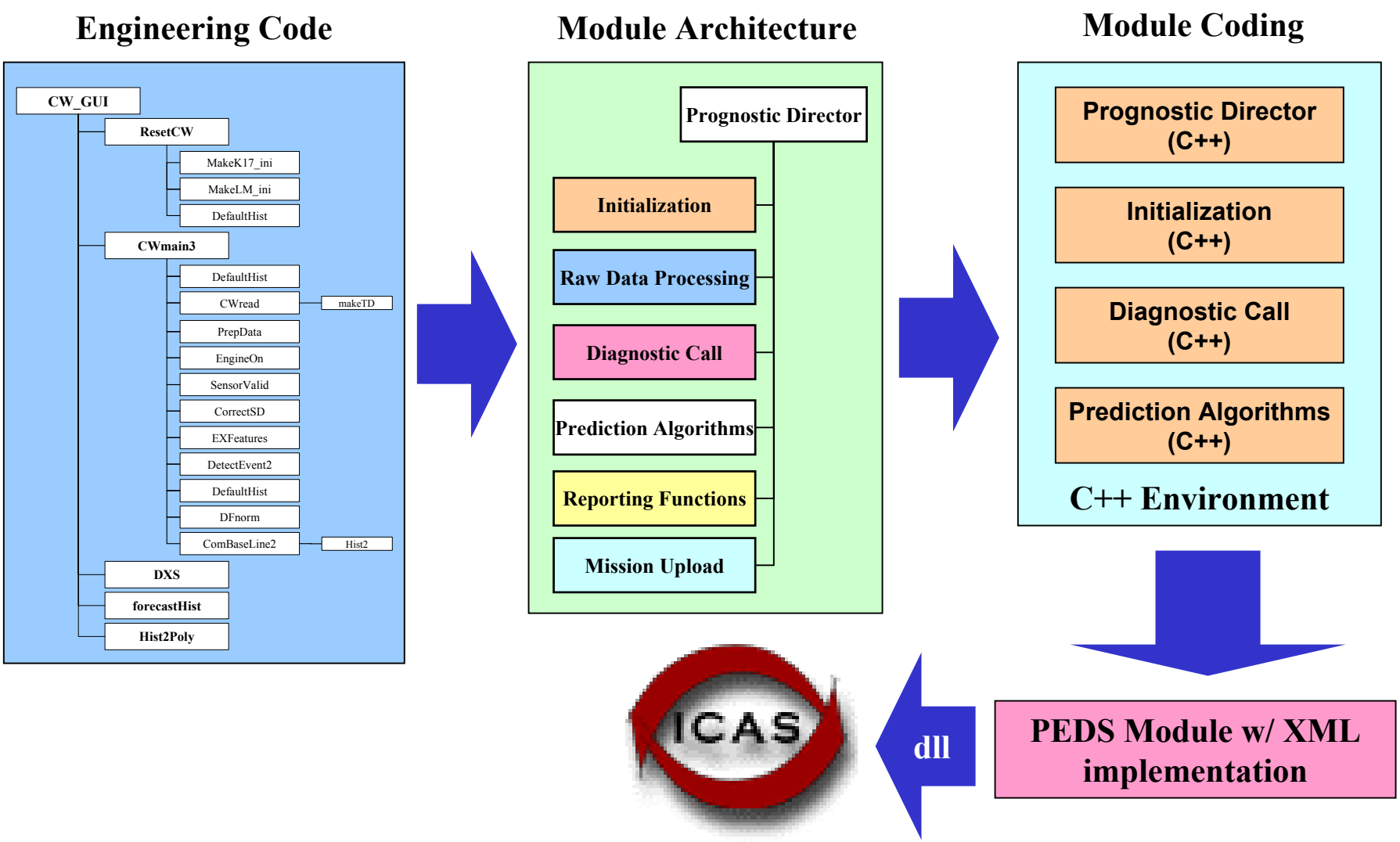
Gas Turbine Generator Performance Prognostics



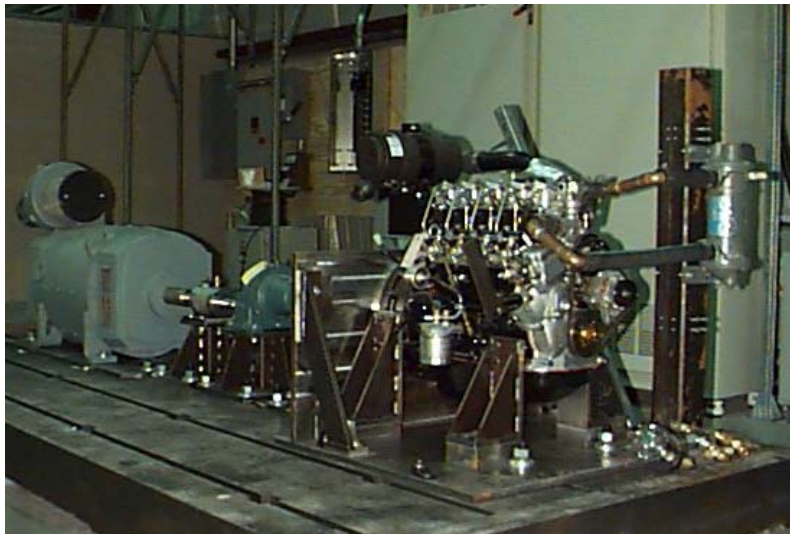


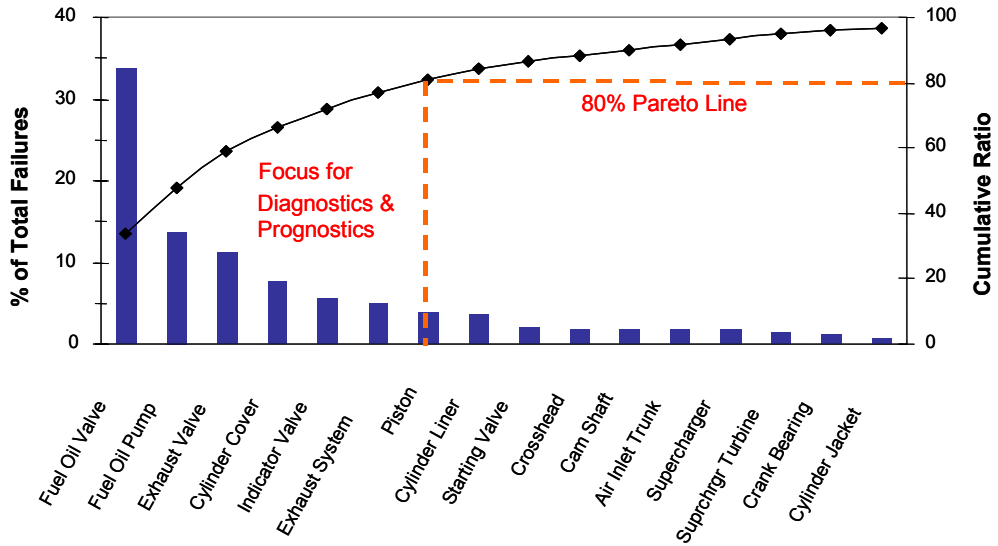


Gas Turbine Performance Prognostics Module



Diesel Engine Prognostics



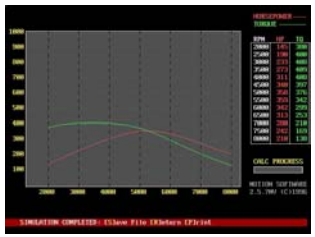


- **Shaft Torsional Vibration:** using high resolution encoders and torque cells
- **Cylinder Pressure Analysis:** monitor combustion process using dynamic pressure
- **Vibration Analysis:** extract features from seeded fault data sets
- **Performance Analysis:** flows, temperatures, and temperatures

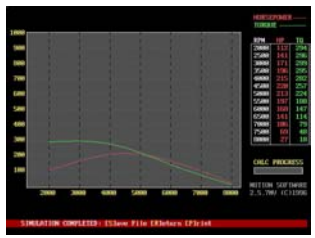
Diesel Engine Performance Prognostics Module

Parameter Shifts over Time

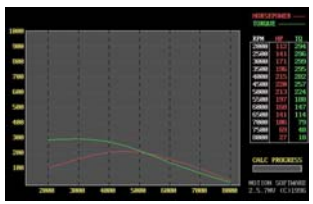
HP



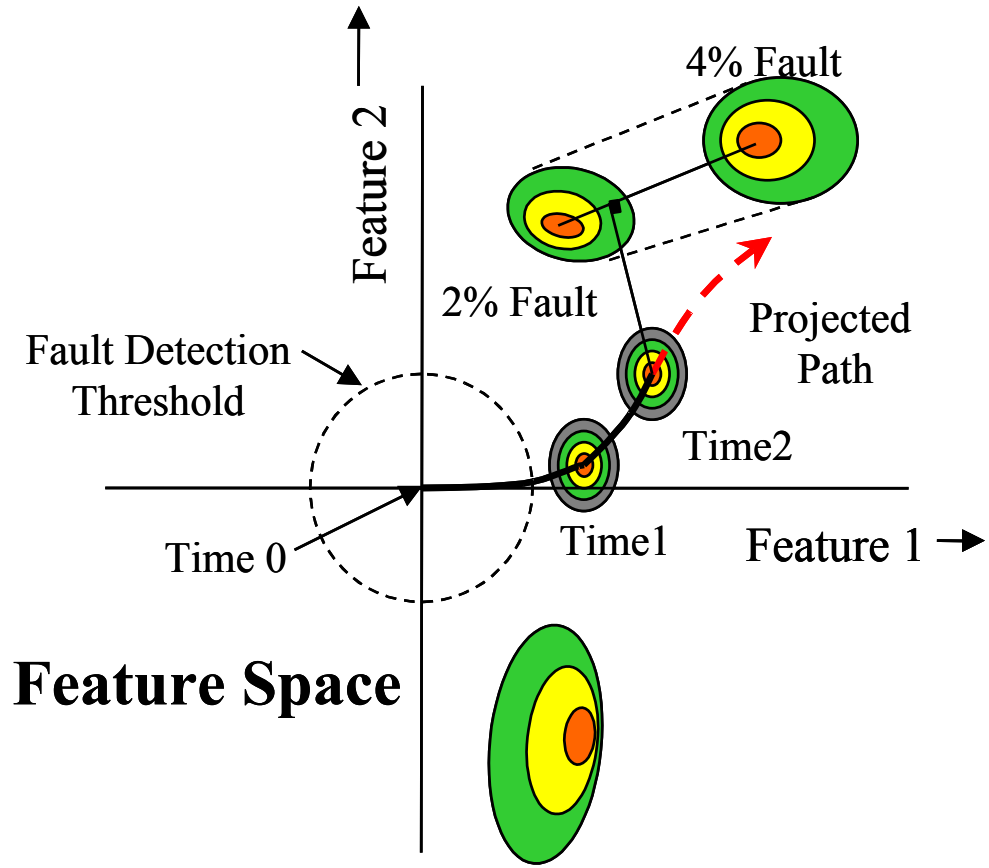
TQ



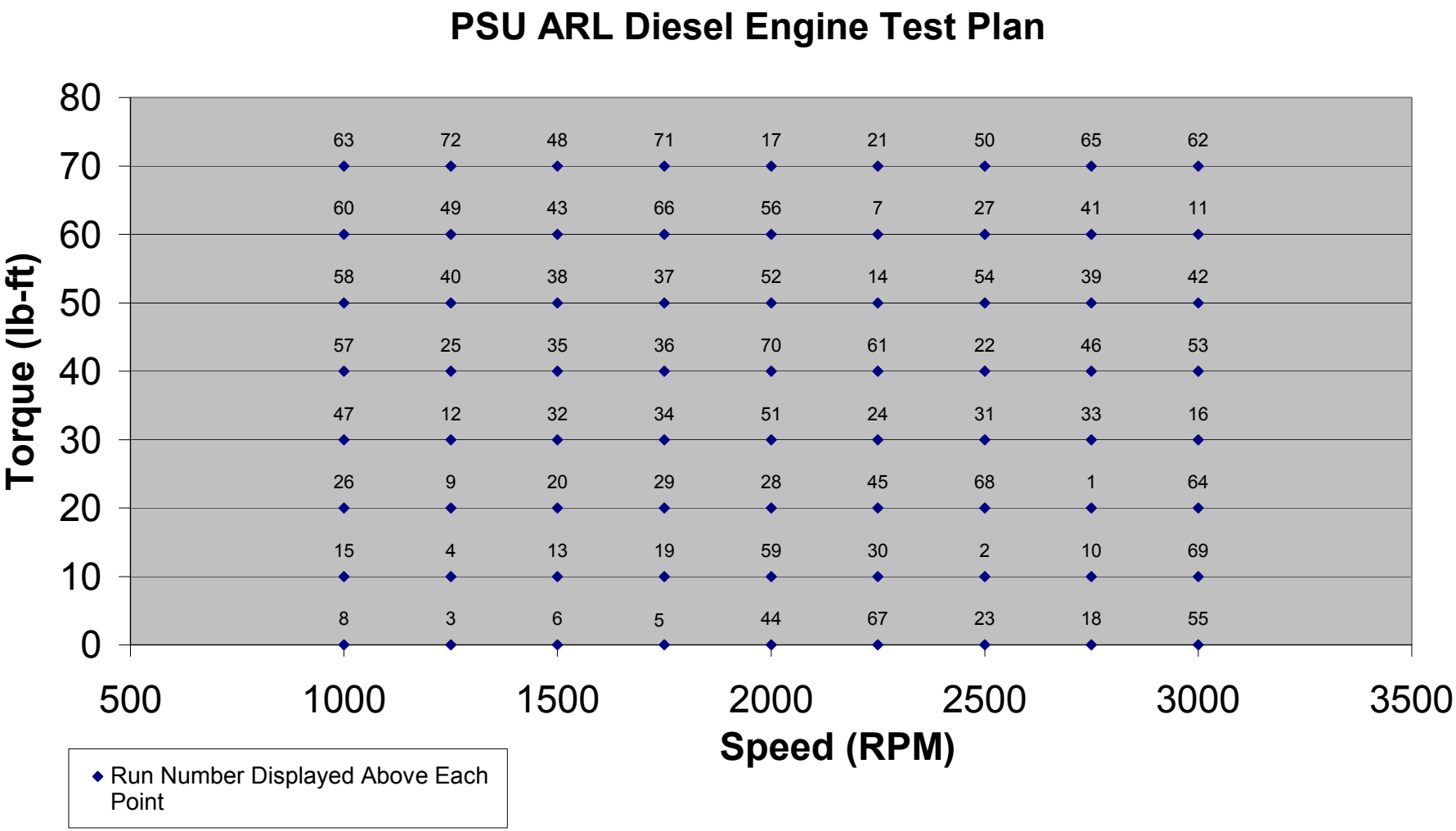
RPM



Track and Predict Path

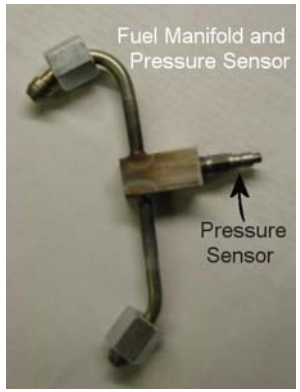


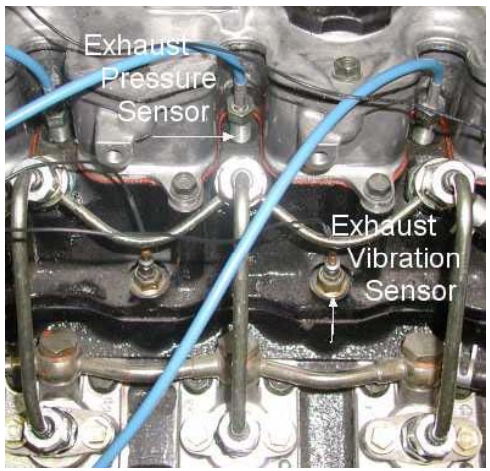
Developing Performance Map

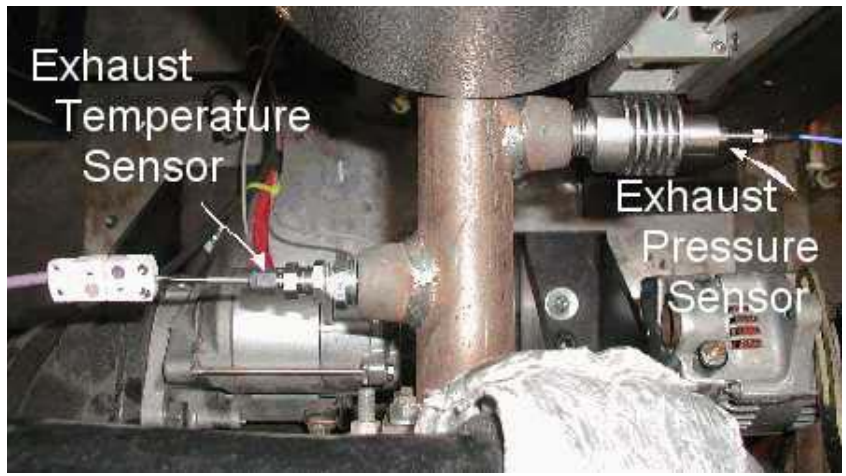


Selected Seeded Fault Tests

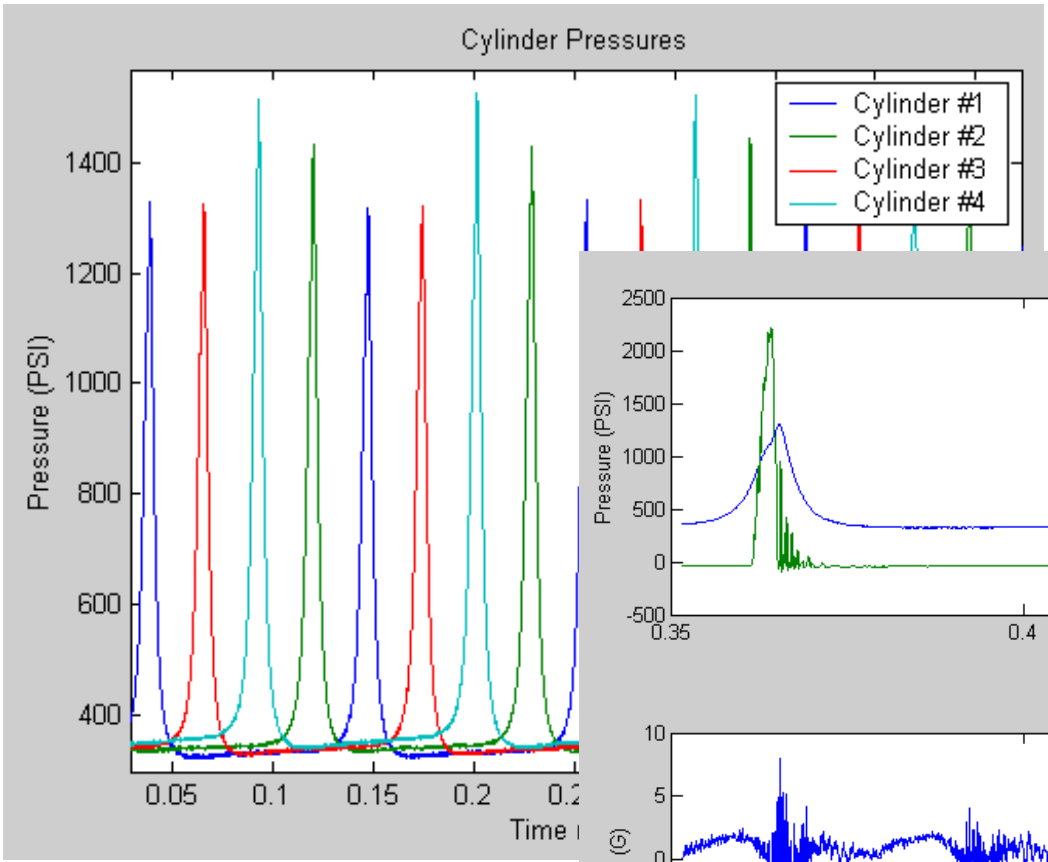
1. Clogged Injector
2. Loss of Compression
3. Intermittent Fuel Pump

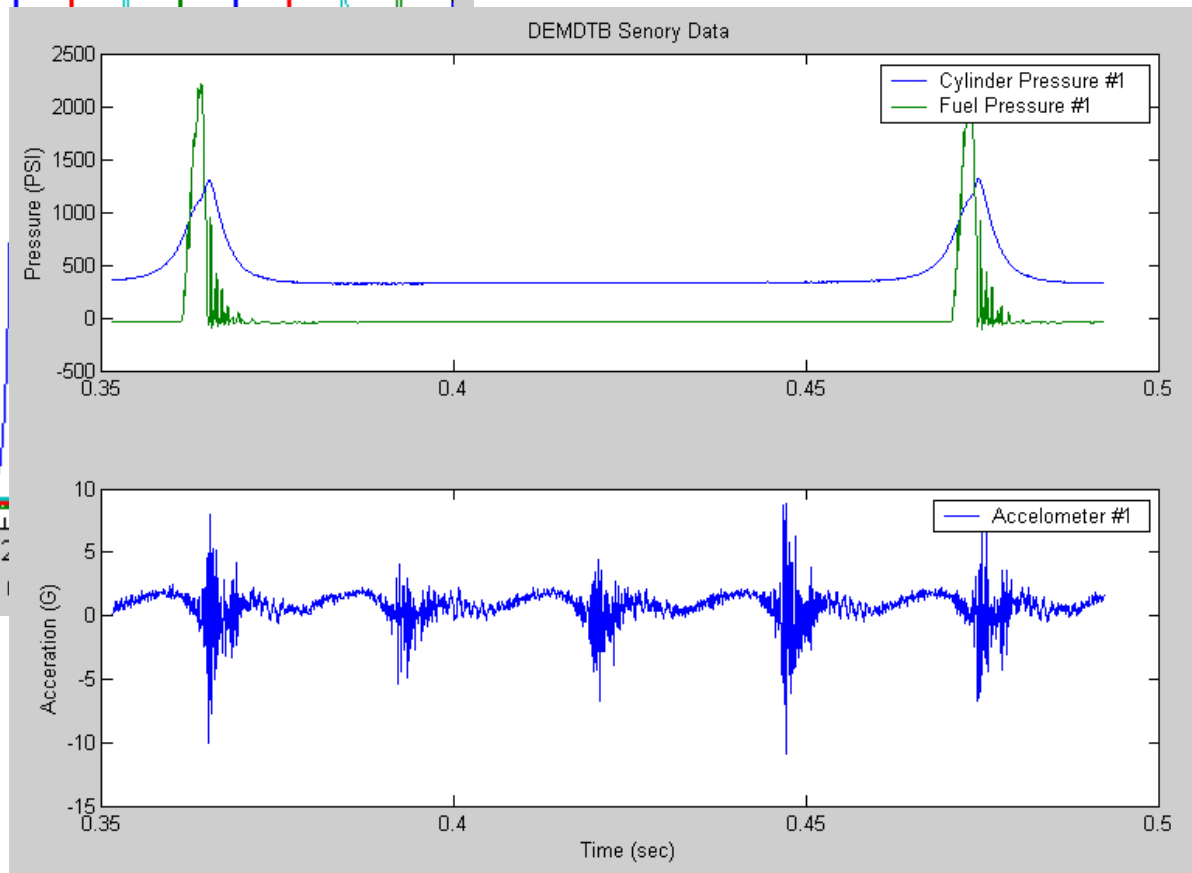






DEMDTB Sensory Data





Module Integration and Demonstration

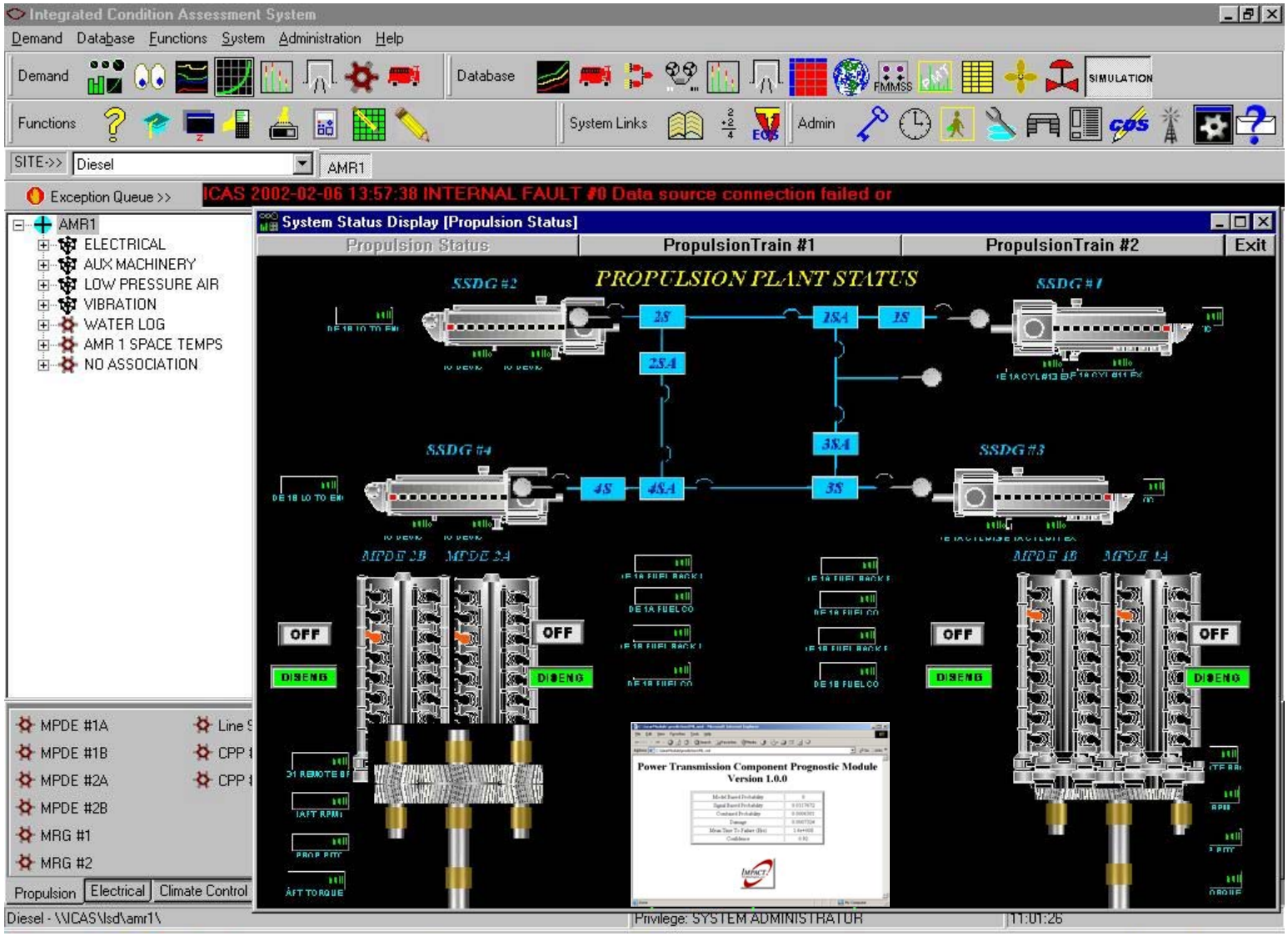
Transition Opportunities

- ICAS Updates
- Gas Turbine CBM Program
- Smart Component/Machinery Efforts
- Machinery and DEC Upgrades

ICAS Integration

- Obtained 2 Configuration Data Sets (CDS)
  - Diesel Engine (LSD-45) and Gas Turbine (CG-47)
  - Still need a few data sets to populate for demo
- Planned ICAS Training Course with Navy personnel (Chris Savage/Russ Leinbach)
- Plans to obtain Demand Data Interface Document (Chris Savage)
  - Explore requirements to interface with ICAS
  - Set-up Prognostic modules as Pseudo-sensors

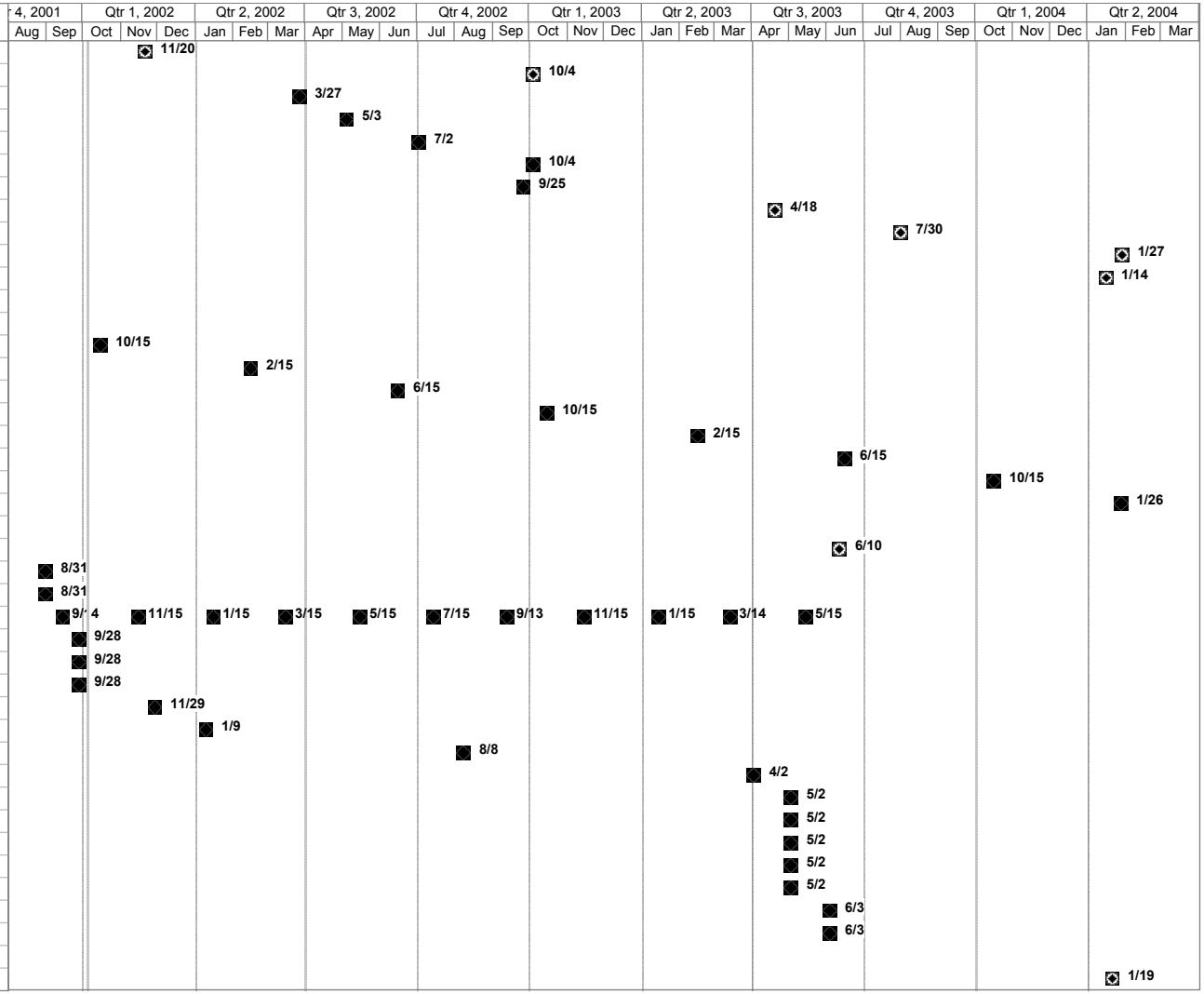
ICAS Integration



Deliverable Status and Near-Term Milestones

Milestone Chart

WBS	Task Name
1	**Complete Task 1 - Mature Generic Prognostic Modeling**
2	**Complete Task 2 - Application Specific Prognostic Algorithms**
2.1	Complete Gas Turbine Performance Prognostic Algorithms
2.2	Complete Gas Turbine Components Prognostic Algorithms
2.3	Complete Power Transmission Component Prognostic Algorithms
2.4	Complete Diesel Engine Comp/Perform. Prognostic Algorithms
3	**Complete Task 3 - Integrate Data/Knowledge Fusion**
4	**Complete Task 4 - Prognostic Module Coding Development**
5	**Complete Task 5 - Testing & Verification of Prognostic Modules**
6	**Complete Task 6 - Implementation in NSC/LBES Application**
7	**Complete Task 7 - Complete HSI development and OSA Tool**
8	**Progress Management Meetings and Demonstrations**
8.1	Progress Management Meetings and Demonstrations 1
8.2	Progress Management Meetings and Demonstrations 2
8.3	Progress Management Meetings and Demonstrations 3
8.4	Progress Management Meetings and Demonstrations 4
8.5	Progress Management Meetings and Demonstrations 5
8.6	Progress Management Meetings and Demonstrations 6
8.7	Progress Management Meetings and Demonstrations 7
8.8	Progress Management Meetings and Demonstrations 8
9	**CDRL Deliverables and Documents**
9.1	Software Development Plans
9.2	Demonstration Project Plan
9.3	**Bi-monthly Status Report**
9.4	Hardware Design Document
9.5	Software Design Document
9.6	System/Subsystem Specification
9.7	Software Requirements Specification
9.8	Preliminary Design
9.9	Interim Report
9.10	Preliminary Report
9.11	Final Hardware Design Document
9.12	Final Software Design Document
9.13	Final Software Requirements Specification
9.14	Final Software Developmental Plans
9.15	Final Demonstration Project Plan
9.16	Drawings & Associated Lists
9.17	Final Report
10	**Final Report for Option Period**



Status and Upcoming Events

- Working on specific tasks in WBS 1 and 2
  - Matured architecture and implementation approach through prototype development – PTC Module
  - Implementing GTP module as a shared library using leveraged GT CBM development
  - Initiated DE data collection and module approach
  - Investigating ICAS demonstration and other transition opportunities
- Obtain HPT Component Information for Modeling
- Attended OSA CBM Last Milestone Meeting in December 2001
- Met with Jim Spaulding regarding diesel engine testing and followup with Terry Foldes on diesel engine trend analysis program

Status and Upcoming Events (con't)

- Completed first 10 CDRL Deliverables and Documents (WBS 9)
  - Software Development Plan (A008)
  - Program Plan (A006)
  - System/Subsystem Design Description (A011)
  - Software Design Description (A010)
  - System/Subsystem Specification (A007)
  - Software Requirements Specification (A009)
  - Developmental Design Drawings and Associated Lists (A012)
  - Status Reports #1-#3 (A001)

Upcoming Conference Publications

- IEEE Aerospace Conference
  - "Prognostic Enhancements to Diagnostic Systems for Improved Condition-Based Maintenance "
  - March 9-16, 2002
  - Big Sky, MT
- 56th Machinery Failure Prevention Technology (MFPT)
  - "Integrating Prognostic Modules Into Existing Naval CBM Systems "
  - April 15-19, 2002
  - Virginia Beach, VA
- Maintenance And Reliability Conference (MARCON)
  - PEDS Session with 5 papers
  - May 5-8, 2002
  - Knoxville, TN

Summary of Phase II Plans

- Prognostic Module Development for Specific PEO-TSC Applications.
  - Feature-Based and Model-Based Integrated Architecture
  - Prognostic Module Tools and Processes
  - Specific Testing for Diesel Prognostics Development
  - Implementing using OSA CBM Functional Hierarchy
- Identify and Integrate into Selected Naval Ship Application(s).
  - Customize Methodology for Specific Application Integration
  - Integration as ICAS Module or Other Specified Platform
  - Utilize Test and Field Data for Validation & Calibration
  - Build upon existing or use Separate Web-based HSI Concept
- Document Methods, Processes, Lessons Learned for Enhancing Existing Navy Diagnostic Systems